Approche pour la construction de modèles d'estimation réaliste de l'effort/coût de projet dans un environnement incertain : application au domaine du développement logiciel

Safae Laqrichi

THÈSE

en vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

délivré par

L'École Nationale Supérieure des Mines d'Albi-Carmaux conjointement avec l'INSA Toulouse

---

présentée et soutenue par

Safae LAQRICHI

le 17 décembre 2015

Approche pour la construction de modèles d'estimation réaliste de l'effort/coût de projet dans un environnement incertain : application au domaine du développement logiciel

---

École doctorale et discipline ou spécialité :
EDSYS : Génie Industriel 4200046

Unité de recherche :
Centre Génie Industriel, Mines Albi

Directeur(s) de Thèse :
M. Didier Gourc, Professeur, Mines Albi
M. François Marmier, Maître assistant HDR, Mines Albi

Autres membres du jury :
Mme Valérie Botta-Genoulaz, Professeur, INSA Lyon, *(Rapporteuse)*
M. Farouk Toumani, Professeur, Université Blaise Pascal-Clermont-Ferrand II, *(Rapporteur)*
M. Xavier Boucher, Professeur, Mines Saint-Etienne, *(Président)*
M. Patrick Paroubek, Ingénieur de recherche,LIMSI, *(Examinateur)*
M. Patrick Hamon, Estimancy, *(Invité)*

À la mémoire de mon père.

À ma chère mère.

À ma famille.

À tous ceux et celles qui m'ont soutenu et qui ont cru en moi ...

Remerciements

Je tiens tout d'abord à remercier Allah, le tout puissant, de m'avoir donné la force et la foi d'arriver à terme de ce travail.

La thèse représente des heures de travail acharné et de rigueur, elle n'aurait pas été possible sans le bienveillant soutien et support de certaines personnes.

Mes premiers remerciements vont d'abord à mon directeur de thèse M. Didier Gourc. Merci pour sa patience, sa rigueur scientifique et sa clairvoyance qui ont été les moteurs de mon travail. Je remercie aussi vivement mon co-directeur de thèse M. François Marmier pour son dynamisme et sa passion qu'il a sut me communiquer.

Je souhaiterais remercier Mme. Valérie Botta-Genoulaz et M. Farouk Toumani, qui m'ont fait l'honneur de rapporter mon manuscrit de thèse avec un regard critique, pertinent et constructif. Merci à M. Xavier Boucher, qui a accepté d'être mon président de jury, et dont l'énergie a illuminé ma soutenance. Merci à M. Patrick Paroubek et M. Patrick Hamon d'avoir accepté de participer à mon jury de thèse malgré leurs emplois du temps surchargés.

Je remercie toutes les personnes formidables que j'ai rencontrées dans l'école des Mines d'Albi et plus particulièrement dans le Centre Génie Industriel. Je pense à Isabelle, Shadan, Laura, Romain, Aurélie, Alexandre, Teixin, Frédérick, Daouda, Frank, Élise, Michel, Jacques, Matthieu, Sébastien, Lionel, Elyes... Merci pour votre gentillesse et votre sympathie. Un grand merci à Nicolas pour m'avoir aidé à apprendre à développer en Java toujours dans la bonne humeur. Merci également à Paul de sa bienveillance et de m'avoir apporté le support nécessaire dans l'utilisation du LaTeX.

Je tiens à remercier également l'équipe de Projestimate, en particulier M. Jean Nevoux et M. Patrick Hamon pour les discussions scientifiques riches que nous avons échangées.

Enfin, les mots les plus simples étant les plus forts, j'adresse toute ma gratitude à ma famille et en particulier à ma mère qui a su m'insuffler le courage et la volonté de me dépasser. Très humblement, je voudrais te dire merci pour ton amour, ta tendresse et tes sacrifices. Que dieu te garde, et te procure santé, bonheur et longue vie.

Merci à ma sœur Charifa pour ses encouragements et son support, en particulier durant les trois derniers mois de rédaction du mémoire où elle a été présente à mes cotés. Merci à ma petite sœur et mon petit frère adorés, Nouhaila et Adam. Je vous aime.

Pour terminer ces remerciements, j'adresse un retentissant hommage à mon feu père Allah y rahmou (paix à son âme), qui n'a pas vu l'aboutissement de mon travail mais je sais qu'il aurait été très fière de sa fille !

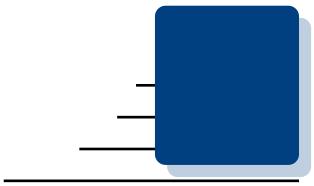
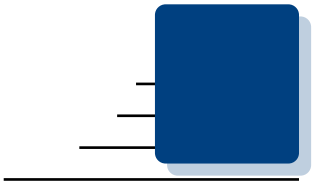
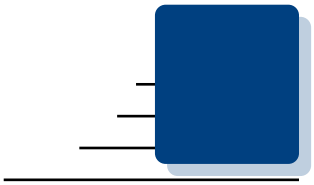


Table des matières

Table des figures



Liste des tableaux



Introduction, contexte et problématique

Introduction générale

Aujourd'hui, les logiciels sont devenus indispensables pour les entreprises et organisations. Ils sont utilisés pour effectuer différentes activités relatives à la gestion des projets et des opérations, la communication, la simulation, etc. Le secteur du logiciel et des services informatiques constitue l'un des plus importants secteurs économiques avec, en 2014, un chiffre d'affaires global de 49,5 milliards d'euros et 365 000 salariés.

Avec l'évolution des industries et de leurs besoins, les logiciels sont devenus de plus en plus grands et complexes. Cependant, le taux de réussite des projets de développement de ces logiciels n'a pas progressé car ils sont devenus plus difficiles à gérer et à réaliser.

Selon une étude menée par le Standish Group International en 2012 (Chaos Report 2012 du Standish Group) [International2013] sur 50.000 projets logiciels, 43% des projets coûtent plus cher et durent plus longtemps que prévu. L'échec et l'abandon des projets logiciel avant leurs achèvements font perdre aux organisations d'énormes sommes d'argent. A titre d'exemple, en 2004, les échecs des projets logiciels ont couté 142 billions d'euros à l'union européen.

L'échec de projets de développement logiciel est dû à plusieurs causes, mais la mauvaise gestion de projets en est la cause majeure. Nombreuses mauvaises pratiques de gestion conduisent à ces

échecs tel que les changements de spécifications, les problèmes de négociation contractuelle et le manque de communication. Cependant, la mauvaise estimation est considérée comme l'un des aspects les plus importants et communs de la mauvaise gestion de projets [Humphrey2001].

L'estimation de l'effort, de la durée et du coût de développement des projets logiciels est une pratique fondamentale dans le management des projets logiciels. Elle est utilisée dans le processus d'appel d'offre ainsi que tout au long du cycle de vie des projets logiciels. Elle permet d'établir un prévisionnel du coût et de la durée du futur projet de développement logiciel en se basant sur les informations disponibles. Cette estimation est cruciale pour les managers de projet dans la négociation des contrats avec le client et dans le pilotage du projet (répartition des ressources, établissement des calendriers, etc.). Elle permet d'aider les managers de projet à atteindre les objectifs du projet et à tenir à leurs engagements contractuels en termes de qualité de logiciel, de budget et de délai. Une mauvaise estimation peut conduire, en plus de l'échec ou l'abandon du projet, à une perte d'argent et des ressources, et une perte de confiance des clients et de l'équipe de développement.

L'histoire des développements logiciels est pleine de cas de projets de développement logiciels qui ont échoué à cause de mauvaises estimations de ressources et de délais. Un exemple très éloquent et récent concerne le projet "Programme national pour la technologie de l'information" (NPfIT) entrepris par le système de la santé publique du Royaume-Uni, le National Health Service (NHS) en 2002. Il a été initialement prévu que le NPfIT coûterait 2,3 milliards £ sur trois ans. En juin 2006, le coût total a été ré-estimé par le Bureau national d'audit à 12.4 milliards £ avec une durée de réalisation de 10 ans. Les responsables impliqués dans le programme ont déclaré estimer le coût final à 20 milliards £, indiquant un possible dépassement de coûts de 440% à 770%. Plus tard, le NPfIT a fini par être abandonné après 12.4 milliards £ de dépenses, des années de retards, des difficultés techniques et des litiges contractuels. Quelques analyses ont statué que les architectes du projet et les fournisseurs ont sous-estimé la complexité du NHS et la puissance de sa communauté clinique [CampionAwwad+2014]. Par conséquent, ils ont sous-estimé l'effort et les coûts nécessaires à la réalisation de ce projet.

L'estimation est une activité complexe qui implique différentes pratiques, de nombreux acteurs (client, directeur, chef de projet, développeurs, etc.) et un environnement dynamique et incertain (dû aux changements des exigences du client, à l'évolution des compétences des développeurs, à la disponibilité des ressources, etc.). Ces éléments sont des facteurs d'échec principaux de l'estimation de l'effort, coûts et durée de projets logiciels.

Le domaine de l'estimation de l'effort de développement de projets logiciels est riche en méthodes et modèles d'estimation qui ont été élaborés et améliorés afin d'obtenir des estimations d'effort de plus en plus réalistes. Les études ont montré que, pour produire des bonnes estimations, les entreprises doivent baser leurs modèles d'estimation sur leurs performances et expériences, notamment représentées par leurs bases de données historiques des projets logiciels réalisés et accomplis. Selon le modèle d'estimation sélectionné pour une organisation spécifique, les résultats et la pertinence des estimations diffèrent. Face à la multitude des modèles d'estimation existants [1], tels que la régression linéaire multiple et les réseaux de neurones, la question qui se pose est alors souvent de savoir quel modèle d'estimation est le plus adéquat.

Nombreuses études ont tenté de répondre à cette question en menant des comparaisons de l'ensemble des modèles les plus utilisés. Les résultats ont conduit à la conclusion qu'il n'existe pas de méthode ou modèle d'estimation performant pour tout contexte et environnement. De surcroit, quelques comparaisons de modèles d'estimation sur le même contexte (c'est à dire même base de données) ont montré des résultats contradictoires et divergents. Les raisons de ces contradictions sont relatives aux techniques d'évaluation et comparaison de modèles d'estimation, qui s'avèrent être souvent non fiables.

1. Un modèle d'estimation, dans ce contexte, est équivalent à une technique de modélisation.

Les incertitudes sont inhérentes à l'environnement et le processus de développement de projets logiciels, en particulier à l'activité d'estimation. Les sources de ces incertitudes sont relatives aux modèles/méthodes d'estimation utilisés et aux données impliquées à ces modèles et méthodes, et aux nouveaux projets à estimer. D'une part, les méthodes et modèles d'estimation ne sont jamais parfaits. Tout d'abord, du fait qu'ils ne sont que des abstractions d'une réalité plus complexe, et ensuite en raison de l'utilisation d'informations et données historiques incertaines dans leurs construction. D'autre part, l'estimation de l'effort de nouveaux projets se base sur les informations disponibles sur ce projet et qui sont naturellement abstraites, incomplètes et imprécises surtout en phases amont. Ainsi, ces incertitudes doivent être représentées dans l'estimation de l'effort, des coûts et de durée.

De ces constats, trois questions ou problématiques majeures émergent :

Q.1 Comment sélectionner, d'une manière fiable et en tenant compte des incertitudes, un modèle d'estimation pour le contexte particulier d'une organisation ?

Q.2 Comment construire un système d'estimation réaliste ?

Q.3 Comment utiliser le système construit pour les estimations de l'effort de nouveaux projets présentant des incertitudes ?

Les travaux de cette thèse cherchent à répondre à ces trois problématiques en proposant une approche générale et structurée permettant d'assister les estimateurs et chefs de projets dans l'estimation réaliste de l'effort de projets logiciels.

Cette thèse a été menée au sein de l'axe ORKID (Organisation, Risque, Connaissance en Conception) dans le laboratoire Centre de Génie Industriel (CGI) de l'école des Mines Albi. Elle s'inscrit dans le cadre du projet FUI 13 Projestimate qui vise à améliorer les pratiques d'estimation, à réduire leurs coûts et à les implémenter au sein d'une plateforme Web Open Source.

Dans la suite de ce chapitre, nous présentons le contexte général de cette thèse.

Généralités sur les projets de développement logiciel

Comme tout projet, les projets logiciels ou projets de développement logiciel consistent en un ensemble d'activités et actions qui conduisent à des résultats. Le résultat principal dans le cas de ces projets est un logiciel/application.

Projets de développement logiciels vs projets de construction ou de fabrication

Un projet de développement logiciel diffère d'un projet d'industrie de construction ou de fabrication à plusieurs égards, entre autres :

— Les logiciels sont des produits intangibles qui sont basés sur des instructions, des formules, des fichiers, etc. alors que les produits des projets de construction sont tangibles et basés sur des éléments physiques.

— Les technologies de développement logiciels évoluent plus rapidement que les technologies (machines, outils, etc.) utilisées dans les autres types de projets d'ingénierie.

— Les projets de développement présentent une bonne flexibilité. Les interfaces et composants d'un logiciel peuvent être changés à toute phase du cycle de vie du projet. Ainsi, les projets de développement logiciel présentent un grand degré de variabilité qui peut rendre difficile l'achèvement du projet dans les coûts et les délais prévus.

— Quand un projet de construction (d'un bâtiment par exemple) est démarré, son avancement est visible sur le chantier, étant donné qu'il s'agit d'un produit physique. Ceci n'est valable pour le cas des projets de développement logiciel que si des méthodes agiles sont employées.

Ces différences sont synthétisées dans le tableau 1.

TABLE 1 – Projets de développement logiciel vs projets de construction ou de fabrication

Critère	Projets logiciel	Projets de construction
Nature du produit	intangible	tangible
Évolution des technologies	rapide et importante	moyenne
Flexibilité et variabilité	importantes	limitées
Avancement projet	Invisible	visible

Facteurs d'échec des projets logiciels

De nombreuses études ont été conduites en vue de déterminer le taux d'échec des projets logiciels et de fournir une vision claire sur l'évolution du domaine de développement logiciel. Le rapport de Standish Group de 2012 indique que seulement 39% des projets sont réussis (livrés dans le temps, en budget et avec toutes les fonctionnalités requises), 43% sont en difficulté (en retard, avec un dépassement de budget et avec moins de fonctionnalités requises) et 18 % échoués (annulés ou livrés mais jamais utilisés). Ces résultats ont été critiqués par quelques auteurs comme étant basés sur des méthodes d'enquête qui sont soupçonnées d'être biaisées [Glass2006]. L'enquête de [El Emam+2008] sur les projets logiciels développés en 2005 et 2007 indique que le taux de réussite des projets logiciels varie entre 48% et 55% et que le taux d'échec varie entre 11.5% et 15.5%. Cette enquête, contrairement aux rapports du Standish Group, conclut qu'il n'y a pas d'importante amélioration entre les deux années 2005 et 2007 et que ce taux reste pourtant élevé. Il est à noter ce taux d'échec est plus important dans le domaine du développement logiciel que dans d'autres domaines tels que l'industrie et la construction [Chemuturi2013].

L'échec des projets logiciels est dû à différentes et nombreuses raisons. Le Standish Group décrit les dix plus importantes, elles comportent entre autres :

- L'incomplétude des exigences et spécifications,
- Le manque l'implication du client,
- Le manque de ressources,
- L'estimation non réaliste,
- Le manque du support exécutif.

De son coté [El Emam+2008] indique dans son enquête que les deux plus importants facteurs d'échecs sont relatifs aux changements des exigences et du cadre du projet, le manque d'implication de la haute direction et le déficit budgétaire dû entre autre à la mauvaise estimation.

Une autre étude de [Humphrey2001] identifie différents facteurs d'échec de projets logiciels dont les planning non réalistes, les effectifs non adéquats et les spécifications changeantes et évolutives sont les plus influents.

Clairement, tous les spécialistes et professionnels sont d'accord que l'un des plus importants facteurs d'échec des projets logiciels est relatif à l'estimation de l'effort. La question se pose concernant les enjeux de l'estimation de l'effort et son rôle dans le management des projets de développement logiciels.

Enjeux de l'estimation

Capers Jone's, spécialiste dans les méthodologies d'ingénierie logicielle, définit l'estimation dans le cadre des projets logiciels comme étant le processus permettant de prédire les réalisations futures d'un projet en termes de différents facteurs, à savoir la taille ou la complexité, le planning, l'effort, les coûts, les bénéfices et le risque [Capers2006].

Il est important de différencier les estimations des objectifs et engagements dans le cadre d'un projet logiciel. En effet, un objectif d'un projet représente un ou plusieurs buts commerciaux

escomptés par le client. Même s'ils sont fortement souhaitables et parfois indispensables, les objectifs d'un projet ne sont pas toujours réalisables. Ils sont souvent établit indépendamment des estimations. Un engagement est une promesse de livrer les fonctionnalités définies conformément à un niveau spécifique de qualité à une date fixée. Un engagement (sur la durée du projet par exemple) peut être identique à l'estimation (de la durée du projet, par exemple), ou il peut être plus agressif ou plus modeste. Autrement, l'engagement n'est pas forcément identique à l'estimation [McConnell2006].

Importance de l'estimation

La bonne estimation de l'effort, des coûts et des délais est un facteur clé de succès des projets logiciels. Par contre, une mauvaise estimation peut avoir des conséquences négatives [Clayton2014], notamment :

— La perte de confiance des clients et de l'équipe dans l'organisation.

— Le mécontentement de l'équipe, en particulier lorsqu'elle doit travailler des heures excessives pour respecter une date de livraison.

— La mauvaise qualité du logiciel due à l'empressement des développeurs à achever le projet et à respecter les délais au détriment des pratiques de contrôle de la qualité, comme les tests.

— L'échec du projet suite aux nombreuse estimations infructueuses et perte totale de la confiance du client dans la capacité de l'équipe à achever et livrer le projet en respectant les engagements.

Estimation et cycle de vie d'un projet

Généralement, l'estimation de l'effort est effectuée et révisée plusieurs fois durant le cycle de vie d'un projet de développement logiciel :

— En amont du projet : Dans la phase de lancement du projet en interne, côté maitrise d'ouvrage, l'estimation est effectuée d'une manière globale et grossière dans le cadre de l'étude de faisabilité. A ce stade, les décideurs s'appuient sur l'estimation globale du projet pour décider de l'approbation du projet et de l'allocation des fonds et ressources. Lors d'une procédure d'appel d'offre, l'estimation est réalisée par les prestataires de développement pour faire des réponses et propositions afin d'acquérir le projet. A ce stade, l'estimation, vue par la maitrise d'œuvre, permet de : (1) assister l'estimateur dans la tarification du projet étant donnée que l'effort de développement est un composant majeur de coût dans le tarif global du projet, (2) estimer les ressources requises pour réaliser le projet, (3) assister les décideurs à déterminer si l'organisation possède les capacités et ressources nécessaires à la réalisation du projet, (4) assister les décideurs à présenter des engagements de livraison aux clients [Chemuturi+2010].

  Généralement, en phase amont du projet, les informations disponibles sur le projet logiciel à développer sont minimales, ainsi la confusion sur l'estimation s'installe. Par conséquent, l'estimation est souvent grossière, pas assez fine et n'est utilisée que d'une façon indicative.

— Durant le développement du logiciel : Tout d'abord, l'estimation de l'effort est utilisée dans la phase de planning de projet comme base pour estimer les ressources requises pour accomplir le projet. Elle est révisée à la fin de la phase de spécification ou/et de la phase de conception afin de valider ou de proposer des modifications des engagements de livraison. L'estimation est utilisée tout au long du processus de développement pour suivre et contrôler l'avancement du projet afin de s'assurer du respect des engagements et prendre les mesures correctives si nécessaire.

  L'avancement du projet apporte progressivement des détails, réduit les incertitudes et lève la confusion sur l'estimation de l'effort.

— Après la livraison du projet : L'estimation de la taille et l'effort du projet sont révisés afin d'analyser la différence entre la taille/ l'effort estimés en phase amont et la taille/ l'effort

estimés (mesurés) à la fin du projet. Cette pratique permet d'avoir un retour d'expérience sur l'estimation et ainsi d'améliorer le processus d'estimation. Malheureusement, cette pratique est peu commune et absente dans de nombreuses organisation [Chemuturi2009].

Facteurs d'échec de l'estimation

L'estimation est une activité complexe qui implique clients et organisations de développement et qui s'appuie sur différents éléments relatifs, entre autres, aux motivations de l'estimation, l'expérience des organisations, et la compétence de l'estimateur. Ainsi, il n'est pas toujours facile et évident de produire une estimation réaliste, fiable et répétable. Plusieurs raisons ou facteurs peuvent expliquer l'échec des estimations, nous citons entre autres :

— Méthodes et modèle d'estimation : Il n'existe pas de méthode ou modèle d'estimation unique performant pour tout contexte et environnement. Les méthodes et modèles d'estimation existants se caractérisent par des degrés variables de succès. Une étude a montré que 30 à 70% des estimations sont incorrectes [Galorath+2006].

— Manque ou mauvaise utilisation des données historiques : Malheureusement, peu d'organisations réalisent la collecte d'informations relatives aux effort, coûts et délais réels après l'achèvement des projets logiciels. Encore moins d'organisation capitalisent les informations sur les projets non réussi et sur les raisons de leurs échecs, notamment les risques auxquels ils étaient confrontés. Les données historiques sont nécessaires pour avoir une vision globale sur le monde réel de développement de projets logiciels. Elles doivent être prises en considération lors de l'estimation, même si elles donnent une vision qui ne satisfait pas les attentes du client et/ou de l'organisation.

— Optimisme excessif du management : L'estimation est souvent conduite par des estimateurs qui manquent de compétence et qui ont tendance à sous estimer les coûts.

— Incertitudes inhérentes au processus de l'estimation : Le développement logiciel est une activité humaine qui est sujette à d'importantes perturbations et incertitudes. Lors de l'estimation, l'estimateur fait nombreuses hypothèses implicites et explicites. Ainsi, l'estimation produite est souvent basée sur ces hypothèses. Les incertitudes sont relatives à multiples composants de l'environnement de l'estimation à savoir les acteurs humains (chef de projet, développeur), la base de données qui peut être erronée et biaisée à cause du manque des informations et en particulier celles relatives aux projets échoués, les techniques et méthodes utilisées (collecte de base de données, modèles d'estimation instables, etc), etc.

— Utilisation d'une forme d'estimation inadéquate : L'utilisation d'une estimation ponctuelle n'est pas pertinente et non représentative de la réalité. Elle ne permet pas non plus de prendre en compte la situation de manière flexible pour pallier aux risques. Une fourchette comprenant une valeur optimiste, une valeur pessimiste et une valeur plus probable est plus adéquate pour refléter les scénarios réalistes du projet et pour un management flexible des risques.

A ces raisons, s'ajoutent souvent des difficultés relatives à l'utilisation de l'estimation et le manque de révision et d'actualisation des estimations tout au long du projet.

Le projet Projestimate (FUI)

Cette thèse s'inscrit dans le cadre du projet Projestimate et est financée par le Fond Unique Interministériel (FUI13). Le projet a été labellisé par le pôle Systematic Paris-Region et est porté par Estimancy. Il implique également divers partenaires industriels et académiques qui sont :

— Deux grandes entreprises/groupes (Banque de France (BdF), PSA Peugeot Citroën),

— Trois petites et moyennes entreprises (Estimancy, ACAPI, Sparkom),

— Un établissement de recherche : Centre génie industriel, Mines Albi à travers Armines

— Un laboratoire public de recherche : LIMSI–CNRS

Objectifs du projet Projestimate

Projestimate a pour objectif principal de fournir aux entreprises une solution permettant de mieux maitriser les projets de développement logiciel grâce à des estimations plus réalistes permettant de mettre en place des indicateurs de pilotage plus efficaces (effort, coût, productivité, etc.). L'enjeu est donc d'améliorer les pratiques d'estimation, de réduire leurs coûts et d'implémenter toutes les méthodes et outils nécessaires au sein d'une plateforme unique supportant les bonnes pratiques de l'estimation (voir figure 1). ProjEstimate fournira aux entreprises une solution permettant :

- De maîtriser leur processus d'estimation tout au long du cycle de vie des projets.
- D'implémenter plusieurs méthodes/modèles d'estimation y compris ceux spécifiques à l'organisation.
- De réduire les coûts d'estimation, en particulier le coût de la mesure fonctionnelle.
- D'obtenir de l'assistance de ses pairs.
- De stocker les différentes estimations d'un projet et les résultats réels dans un référentiel.
- D'exporter les résultats vers des outils de reporting ou de pilotage.
- D'analyser le référentiel pour analyser la performance de l'organisation, se comparer avec l'extérieur et affiner les paramètres d'estimation.

En outre, le projet a comme objectif complémentaire de constituer une communauté ProjEstimate qui permettra :

- L'entraide, le partage des connaissances, des expériences, des idées, le coaching, la formation, etc.
- Le partage anonyme des données pour le Benchmarking.
- L'amélioration permanente du logiciel en stimulant les axes de recherche pour étudier les possibilités de simplifier les estimations, améliorer leur justesse et réduire leurs coûts.

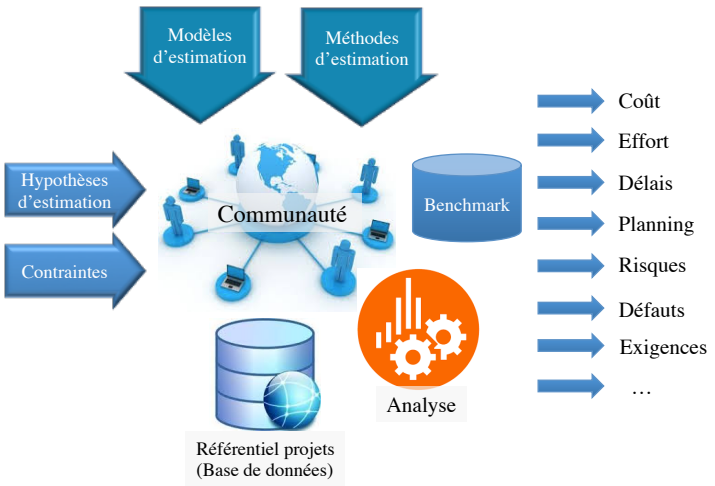


FIGURE 1 – Vue d'ensemble de la plateforme ProjEstimate

Verrous scientifiques de Projestimate

Le processus global de l'estimation logicielle comprend deux activités principales : l'estimation de la taille/complexité et l'estimation de l'effort, coût et durée du projet. ProjEstimate a pour vocation d'améliorer ces deux activités et d'alléger fortement les difficultés liées à leurs complexité. Ainsi, deux verrous scientifiques ont été identifiés et auxquels Projestimate vise à apporter des solutions. Ces verrous ou axes de recherche sont :

- **Verrou n°1** : La semi-automatisation du calcul de la taille fonctionnelle des logiciels et nouveaux modèles d'estimation. Ce verrou est traité par le laboratoire LIMSI-CNRS en s'appuyant sur l'approche sémantique.
- **Verrou n°2** : La mise en place de nouveaux systèmes d'estimation réalistes dans un environnement incertain. Ce verrou constitue la problématique de cette thèse. Les recherches dans le cadre de cette thèse visent à proposer, en expliquant le raisonnement suivi, une approche de construction et d'utilisation d'un système d'estimation en prenant en compte les incertitudes présentes dans le processus d'estimation. Le système d'estimation à construire doit :
  - garantir une bonne précision et performance des estimations,
  - assurer la stabilité des résultats afin qu'il fournisse des estimations assez proches pour des projets sensiblement similaires,
  - livrer une valeur d'effort et un degré de confiance associé,
  - être opérationnel et permettre une facilité d'utilisation.

Organisation du manuscrit

Le plan de ce manuscrit est illustré dans la figure 2. Après ce chapitre d'introduction, la suite de ce manuscrit est composée de 5 chapitres.

- Le chapitre 1 fait l'état de l'art autour de la problématique de la thèse. Dans un premier temps, nous introduisons le processus global de l'estimation. Nous présentons et classifions ensuite les différentes techniques existantes sur l'estimation de l'effort de développement logiciel et en établissons une synthèse. Dans un deuxième temps, nous présentons les incertitudes inhérentes au processus d'estimation ainsi que les approches et pratiques à adopter face à ces incertitudes. A la fin, nous discutons la problématique de sélection de modèles d'estimation d'effort et concluons avec une synthèse des concepts et des questions soulevées dans le cadre de cette thèse.
- Le chapitre 2 répond à la question Q.1. Il s'intéresse à proposer une approche générale et structurée permettant d'assister les entreprises dans la sélection du modèle d'estimation le plus adapté à leurs besoins, contexte et environnement. Cette approche permettra la comparaison fiable de plusieurs modèles candidats et la sélection d'un modèle d'estimation selon une multitude de critères incluant un critère quantifiant et représentant les incertitudes.
- Le chapitre 3 répond aux questions Q.2 et Q.3. Il se focalise dans un premier lieu sur la construction du système d'estimation d'effort réaliste propre à l'entreprise à partir du modèle d'estimation sélectionné en utilisant l'approche exposée dans le chapitre 2. Il présente dans ensuite la procédure d'utilisation du système d'estimation d'effort construit dans l'estimation d'un nouveau projet caractérisé par des incertitudes.
- Le chapitre 4 expérimente et valide l'approche proposée sur cinq différentes bases de données. Quatre modèles d'estimation sont comparés sur les cinq bases de données et les résultats obtenus sont discutés. Nous exposons ensuite, dans ce chapitre, l'application web Estimset implémentée pour automatiser l'approche proposée.
- Le dernier chapitre présente une synthèse globale des travaux de recherches effectués dans le cadre de cette thèse ainsi que les perspectives.

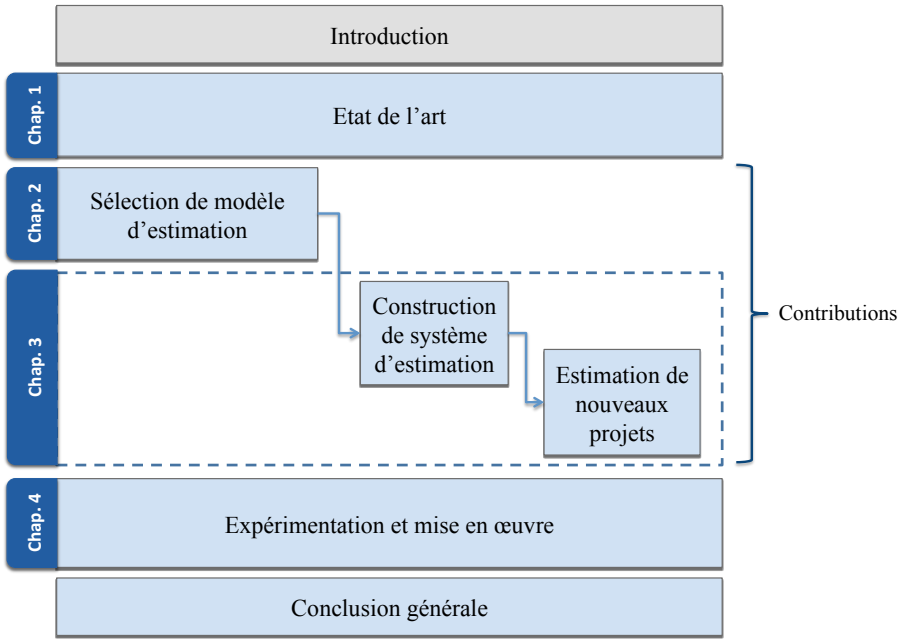
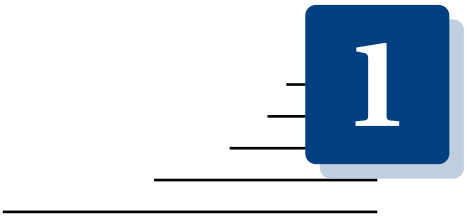


FIGURE 2 – Organisation du manuscrit suivant les activités de l'approche globale proposée

CHAPITRE



État de l'art

Introduction

Après avoir présenté, dans le chapitre précédent, le contexte des travaux de cette thèse, nous dressons, dans ce chapitre, un état de l'art relatif aux techniques d'estimation, aux incertitudes et techniques de sélection de modèle d'estimation.

L'objectif de ce chapitre est de : (1) comprendre les spécificités des différentes techniques d'estimation afin de sélectionner celles qui peuvent répondre à notre problématique, (2) présenter les différents types d'incertitudes rencontrées lors du processus d'estimation et montrer la nécessité de tenir compte de ces incertitudes et (3) étudier les travaux existants sur la sélection de modèle d'estimation et identifier leurs limites et points d'amélioration.

Ce chapitre comporte quatre sections. Nous introduisons, dans la première section, le processus global d'estimation. Nous présentons ensuite, dans la deuxième section, une classification des différentes techniques existantes d'estimation des coûts de développement logiciel. La synthèse de cette section mènera au type de technique d'estimation à retenir dans la suite de ces travaux de thèse. La section qui suit, développe les incertitudes présentes dans le processus d'estimation, ainsi que les approches et pratiques proposées dans la littérature pour face à ces incertitudes. La section 4 discute la problématique de sélection de modèles d'estimation d'effort et met l'accent sur les conflits et contradictions entre les résultats des travaux de comparaison de modèles d'estimation existants dans la littérature. Nous concluons ce chapitre par une synthèse des concepts et des questions soulevés ainsi que des solutions existantes.

1.1 Processus d'estimation

Le processus d'estimation repose généralement sur le processus représenté dans la figure 1.1. Ce processus est composé de deux activités :

— Le calcul ou la mesure de la taille fonctionnelle

— L'estimation (ou la quantification) de l'effort, du coût, de la durée,...

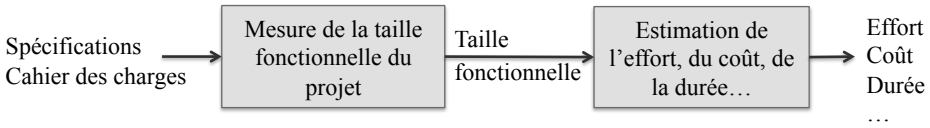


FIGURE 1.1 – Processus d'estimation

1.1.1 Mesure de la taille fonctionnelle du projet

La mesure de la taille fonctionnelle est une activité importante dans le domaine de l'ingénierie logicielle.

La taille fonctionnelle est définie comme la mesure de ce que le système peut faire, en termes de capacité de traitement de l'information du système [Moores2001]. Elle est dérivée de la quantification des exigences fonctionnelles exprimées par les utilisateurs [ISO2007]. En d'autres termes, la mesure de la taille fonctionnelle s'appuie sur les spécifications du projet et du cahier des charges exprimant les exigences et les besoins du client. Elle peut être exprimée dans différentes unités telles que les points de fonction (PF) [1], les lignes de code source (SLOC) [2], les points de cas d'utilisation (PCU) [3], etc.

La taille fonctionnelle constitue une base pour :

— L'estimation de l'effort, du coût et de la durée des projets logiciels en phases amont des projets,

— Le contrôle et le suivi de l'évolution des fonctionnalités tout au long du cycle de vie des projets,

1. Le point de fonction est une unité de mesure standardisée et indépendante de la technologie de programmation utilisée. Elle quantifie les fonctionnalités offertes en les décomposant en des éléments tels que les fonctions d'entrée de données et celles de sortie (restitution) de données [Asadullah2015]. La somme des points attribués à tous les élément donne la taille fonctionnelle.

2. SLOC est le nombre de lignes du code source du logiciel.

3. Le point de cas d'utilisation est une mesure dérivée des exigences du client exprimées sous la forme d'un diagramme de cas d'utilisation. Cette mesure est calculée en s'appuyant sur les éléments des cas d'utilisation et prend en compte des considérations techniques et environnementales.

— La mesure de la productivité des projets de développement logiciel, par exemple en terme de nombre de jours-homme (effort réel) par point de fonction. Par exemple, la productivité calculée à partir d'un projet d'une taille de 50 PF, réalisé en 100 jours-hommes est égale à 20 jours-homme/PF. Cette mesure permet de comparer les projets entre eux et à effectuer une analyse de la performance de l'organisation. Une organisation qui a une productivité moyenne de 25 jours-homme/PF est plus compétitive qu'une organisation ayant une productivité de 20 jours-homme/PF.

Il existe différentes méthodes de calcul et de mesure de la taille fonctionnelle, elles peuvent être classées en deux catégories :

— Méthodes de mesure par jugement d'expert : elles sont informelles, intuitives et se basent sur l'expérience construite à partir des projets similaires réalisés. La mesure de la taille du projet est conduite par un ou plusieurs experts sur la base des spécifications exprimant les exigences et les besoins du client.

— Méthodes de mesure formelles : elles se basent sur des éléments quantitatifs et mesurables caractérisant le projet logiciel. Ces éléments décrivent les aspects importants des exigences et spécifications des projets à développer, telles que le nombre d'exigences et le nombre de transactions [Laqrichi+2013a].

  Ces méthodes peuvent être classifiées en deux catégories : (1) les méthodes de mesure standards qui ont été acceptées par l'ISO (International Organization for Standardization) en tant que norme internationale, telles que IFPUG [IFPUG1999] qui est la version normalisée des PF introduite par Albrecht en 1979 [Albrecht1979], COSMIC FP [ISO2003] et NESMA FP, et (2) les méthodes de mesure non standards développées par des chercheurs ou des entreprises telles que les PCU [Anda+2001] et les unités d'œuvre standard (UO) [4].

La taille fonctionnelle a été identifiée par plusieurs travaux comme variable d'entrée principale de l'activité d'estimation de l'effort [Jorgensen+2007 ; Pfleeger2005 ; Sunkle+2012], de ce fait, une mesure précise de la taille est primordiale pour une estimation précise et réaliste de l'effort.

1.1.2 Estimation de l'effort, de la durée et du coût

L'effort ou la charge de travail est la quantité de travail nécessaire pour accomplir le projet de développement d'un logiciel [Muzaffar+2010]. L'unité de mesure la plus utilisée pour exprimer l'effort de développement de logiciel est le mois-homme. Cette mesure représente le nombre de personnes nécessaire pour réaliser une tâche en une période d'un mois. Concrètement, si le développement d'un projet nécessite le travail de 10 personnes pendant 8 mois, l'effort de développement de ce projet est alors 80 mois-hommes.

Cette définition peut laisser penser que les mois et les hommes sont interchangeables. Dans le cas de notre exemple il est possible qu'une seule personne réalise le projet en 80 mois, et que 80 personnes le réalisent en un seul mois. Ceci est expérimentalement faux. Ce constat a été proclamé la première fois par BROOKS en 1975 dans son livre "Le mythe du mois-homme" [Brooks2001]. En effet, Brooks souligne que les mois et les hommes ne sont interchangeables que dans le cas où une tâche peut être divisée et réalisée par plusieurs personnes sans qu'il y ait besoin de communication entre eux. Or, ce cas est assez rare dans le domaine du développement logiciel. Généralement, les activités de développement d'un logiciel sont séquentielles et ne sont, de ce fait, pas partitionnables.

Par contre, le concept du mois-homme, l'effort, l'effectif et la durée nécessaires au développement d'un logiciel sont liés par la relation 1.1.1. En outre, le coût de développement est aussi fonction de l'effort et peut être calculé du moment où les coûts mensuels des personnes intervenants sont connus (voir équation 1.1.2). Le coût total d'un projet inclue d'autres charges en plus des coûts de développement. Ces charges incluent les acquisitions ou les locations de matériels et de logiciels,

4. L'unité d'œuvre standard est une mesure qui repose sur la décomposition des spécifications ou le cahier des charges du client en des entités de travail techniques à réaliser, par exemple la création d'écran, la modification de table relationnel, etc.)

les frais de déplacements (réunions et essais), les télécommunications (appels, vidéoconférences, etc.), les formations, les frais de locaux, etc. [Chemuturi+2010]. Du fait que les délais et les coûts de développement sont souvent dérivés de l'effort, la terminologie "estimation de l'effort" peut être utilisée pour désigner d'une façon générale l'estimation de l'effort, de la durée ou/et des coûts de développement logiciel.

$$Effort = Durée \times Effectif \tag{1.1.1}$$

$$Coût = coût/\ Unité\ d'effort \times Effort \tag{1.1.2}$$

1.2 Techniques et modèles d'estimation d'effort

Il existe différentes techniques d'estimation de l'effort de développement logiciels. Chaque technique a des spécificités et des caractéristiques relatives aux éléments impliqués tels que les informations sur les projets à estimer et le niveau d'expérience de la personne chargée de l'estimation. Quelques techniques demandent des informations qui sont connues très tôt dans le cycle de vie de projet, par exemple nombre d'écrans connu des spécifications. D'autre techniques nécessitent des informations qui ne peuvent être connues que plus tard dans le cycle de vie de projet, par exemple nombre de personnes de l'équipe. En avançant dans le cycle de vie du projet, les informations deviennent plus certaines. Ainsi, il y a plus de chance de produire des estimations plus pertinentes et réalistes. L'utilisation d'une méthode d'estimation adéquate permet d'obtenir des estimations plus réalistes.

Différentes classifications de techniques d'estimation ont été proposées dans la littérature. Nous proposons une classification de techniques d'estimation sur plusieurs niveaux (voir figure 1.2).

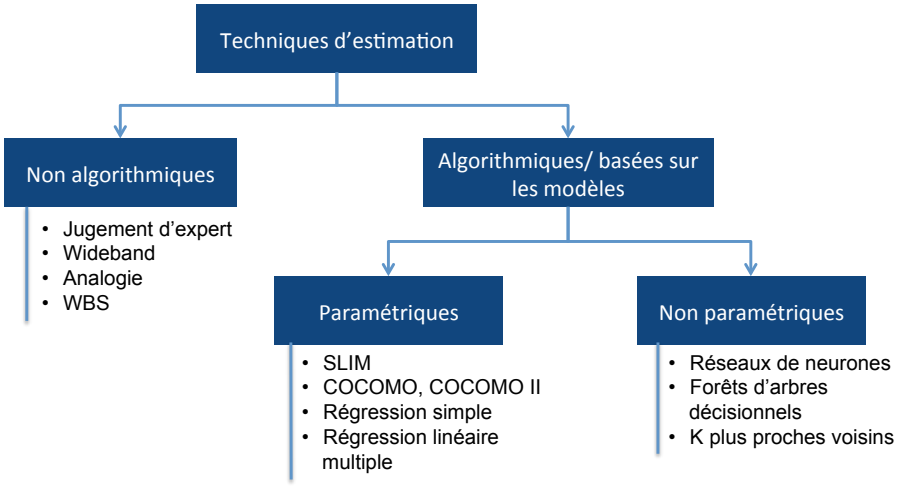


FIGURE 1.2 – Classification des techniques d'estimation d'effort

Le premier niveau les classifie selon le type du processus mental appliqué dans l'étape de quantification, c.à.d. l'étape permettant de traduire la compréhension du problème d'estimation de développement de logiciel en une mesure quantitative de l'effort nécessaire [Jorgensen2007]. Dans ce niveau, ils sont classifiés en :

— Techniques non algorithmiques,

— Techniques algorithmiques [LopezMartin2015].

Ci-dessous, les spécificités de chaque catégorie sont détaillées ainsi que les sous catégories associées aux autres niveaux et accompagnées d'exemples de méthodes d'estimation utilisées.

1.2.1 Techniques non algorithmiques

Cette catégorie (classe) regroupe les méthodes et techniques d'estimation les plus utilisées selon plusieurs revues sur les pratiques d'estimation. Ces méthodes non formelles repose sur l'expérience. Elles sont conduites par une ou plusieurs personnes dites expertes dans le domaine de l'estimation. Elles se basent essentiellement sur l'intuition ainsi que sur l'expérience appuyée par l'historique des projets similaires. Ces techniques sont rapides et peuvent être utilisées assez tôt dans le cycle de développement logiciel. Cependant, elles sont très subjectives, non répétables et manquent d'argumentation analytique [Jorgensen2004a]. Parmi ces techniques, nous pouvons citer le jugement d'expert, l'analogie et l'estimation analytique.

1.2.1.1 Jugement d'expert

Cette technique peut être individuelle ou collective. Lorsqu'elle est exercée de manière individuelle par un seul expert qui estime l'effort et le coût, les résultats sont très dépendants de son expertise. Cet aspect est atténué par une analyse collective comme dans les approches de type Wideband Delphi et planning poker [Briand+2002].
La technique Wideband Delphi cherche à atteindre un consensus entre les estimations fournies par les experts. Dans cette méthode, les experts du groupe estiment la taille ou l'effort du projet d'une façon indépendante, ensuite, discutent la nature des différences d'estimation, puis refont l'estimation jusqu'à ce qu'ils arrivent au consensus [Boehm1981].
Planning poker est une variante de la technique Delphi qui est surtout utilisée pour les projets menés par la méthode agile [MoløkkenØstvold+2008] pour évaluer les fonctionnalités du logiciel appelées scénarios d'utilisateur (user stories). Dans cette technique, chacun des experts utilise une carte numérotée qu'il pose sur la table face vers le bas, au lieu de donner l'estimation oralement. Ceci permet de ne pas influencer l'avis des autres membres du groupe. Les cartes sont ensuite révélées et le groupe discute et décide sur l'estimation à retenir.

1.2.1.2 Estimation par analogie

L'estimation de l'effort nécessaire pour effectuer une tâche peut être faite par analogie avec des tâches similaires déjà effectuées. Cette technique est rapide à mettre en œuvre et peut être utilisée durant tout le cycle de vie du logiciel. L'utilisation de cette technique comporte tout de même quelques difficultés. La première difficulté est de définir le bon critère de mesure de similarité. Une autre difficulté majeure est de trouver dans l'ensemble des tâches réalisées, une tâche suffisamment similaire à la tâche à estimer, c'est à dire une tâche qui présente approximativement les mêmes caractéristiques que la tâche à estimer. Cette difficulté s'accroit pour le cas des entreprises ne disposant pas ou manquant d'expérience dans le domaine du projet étudié, car elles n'ont pas suffisamment d'historique sur lequel elles peuvent se baser afin d'établir des analogies [Goodman1992].
Il est à noter que le concept d'analogie a été modélisé par la technique de raisonnement à partir de cas (RàPC), permettant ainsi de formaliser le processus d'estimation par analogie sous la forme de modèles appelés EbA (Estimation basé sur l'Analogie) [Chiu+2007 ; Keung2009 ; Shepperd+1997].

1.2.1.3 Estimation analytique

L'estimation analytique, également appelée l'estimation ascendante (bottom-up), est la plus ancienne et la plus classique de toutes les méthodes d'estimation. Elle se base sur le découpage du projet en des activités et tâches élémentaires (Work Breakdown Structure). Souvent, l'effort de chaque activité est estimé, généralement par jugement d'expert. L'estimation de l'effort total du projet est alors la somme de l'effort des estimations de chaque activité du projet, éventuellement avec l'ajout d'une quantité d'effort pour couvrir les activités et les événements imprévus [Jorgensen2004b]. Cette méthode nécessite une connaissance détaillée du projet logiciel à développer, elle ne peut en conséquence être appliquée que dans les phases aval du projet.

1.2.2 Techniques algorithmiques

Pour remédier aux limitations des techniques non algorithmiques et informelles, les techniques algorithmiques s'appuient sur la modélisation statistique formalisant les pratiques d'estimation.

Les techniques algorithmiques se basent sur des modèles élaborés à partir de bases de données historiques. Ces modèles expriment les relations mathématiques qui existent entre l'effort et les variables discriminantes considérées déterminantes de l'effort et appelées "conducteurs d'effort" ou "facteurs d'effort".

L'élaboration des modèles d'estimation s'appuie sur une base de données collectées sur les projets réalisés.

La procédure d'élaboration des modèles d'estimation se compose de deux étapes majeures :

- L'identification des facteurs d'effort : les facteurs ayant une influence sur l'effort de développement logiciel sont identifiés par l'expert soit d'une façon intuitive en s'appuyant sur son expérience ou en procédant par analyse statistique de la base de données historique des projets réalisés.

  Comme déjà mentionné dans la section 1.1.1, la taille du projet logiciel est souvent considérée comme facteur d'effort principal. Cependant, il n'est pas unique, d'autres attributs peuvent être identifiés comme facteurs d'effort tels que l'expérience des développeurs ou encore la technologie utilisée.

- L'identification de la relation entre l'effort et les facteurs d'effort : cette relation mathématique peut être représentée par une fonction $\widetilde{f}$ qui vise à représenter la relation réelle f entre l'effort et les autres attributs. Déterminer $\widetilde{f}$ revient tout d'abord à définir son type, en d'autre terme choisir la technique de modélisation à utiliser, telle que la régression linéaire multiple, les réseaux de neurones et les arbres de décision. Ensuite, la forme de $\widetilde{f}$ doit être déterminée par ajustement sur la base de données d'étude. En effet, la fonction f recherchée doit pouvoir produire les estimations de l'effort les plus proches possible des efforts réels. Autrement dit, elle doit permettre de minimiser les erreurs d'estimation ε. La fonction $\widetilde{f}$ est alors de la forme :

  $$\widetilde{f}(x_{i1}, x_{i2}, \cdots, x_{ie}) = \widetilde{E}_i = E_i + \varepsilon_i \quad \forall i \in 1, \cdots, N \tag{1.2.1}$$

  où x_{ij} est la valeur du facteur d'effort j collecté sur le projet i. E_i représente l'effort réel du projet i, $\widetilde{E}_i$ représente l'effort estimé du projet i et e est le nombre de facteurs d'effort.

Divers modèles d'estimation d'effort ont été proposés par des chercheurs et des praticiens au fur et à mesure de l'évolution du développement logiciel. Ils peuvent être classifiés, selon leur nature et les hypothèses sur lesquelles ils se reposent (normalité des données par exemple), en deux catégories :

— Modèles paramétriques : sont des modèles d'estimation ayant une forme bien définie et un nombre fixe de paramètres.

— Modèles non paramétriques : sont des modèles d'estimation ayant une forme non déterminée à priori et un nombre de paramètres non fixé à priori.

Il faut aussi noter que, récemment d'autres modèles ont été développés afin de combiner les avantages des modèles paramétriques, tels que la simplicité, avec les avantages des modèles non paramétriques, tels que la capacité à modéliser des relations complexes. Un exemple de cette catégorie est le modèle d'estimation LSEbA proposé par [Mittas+2010].

Dans les sections suivantes, nous allons décrire ces classifications, leurs spécificités et limitations, ainsi que des exemples des modèles d'estimation de l'effort les plus connus et utilisés de chacune des catégories.

1.2.3 Modèles d'estimation paramétriques

Les modèles paramétriques reposent sur d'importantes hypothèses sur la distribution des données, telles que la normalité. Chaque modèle d'estimation paramétrique s'appuie sur une forme de fonction déterminée à priori et un nombre déterminé à priori de paramètres. Les paramètres des modèles d'estimation paramétriques sont appelés les coefficients. Ces coefficients sont directement déterminés à partir de l'apprentissage sur la base de données. Des exemples courant de modèles paramétriques sont SLIM [Putnam1978], COCOMO [Boehm1981] et la régression linéaire [Brunet2010].

1.2.3.1 Modèle SLIM : Software LIfecycle Management

Développé en 1978 [Putnam1978], SLIM est considéré comme l'un des premiers modèles d'estimation algorithmiques de l'effort de développement logiciel. PUTNAM, l'auteur de la méthode SLIM, s'est basé sur la distribution de Rayleigh [5] et sur des observations concernant le niveau de productivité pour le construire. Il se définit comme suit :

$$\widetilde{E}_{total} = \left(\frac{LOC}{CT^{4/3}}\right)^3 \tag{1.2.2}$$

où $\widetilde{E}_{total}$ est l'effort total du cycle de vie estimé en homme-années, T est le temps de développement, LOC est la taille du logiciel estimée en ligne de code et C est le facteur de technologie.

Le modèle SLIM utilise deux équations pour dériver l'équation 1.2.2. Ces équations décrivent la relation entre l'effort et la durée de développement. La première équation, appelée l'équation du niveau de productivité du logiciel, exprime l'effort de développement en fonction de la taille du projet et le temps de développement. La deuxième équation, appelée l'équation de la main-d'œuvre exprime l'accumulation de la main-d'œuvre en fonction du temps [Fairley1992].

1.2.3.2 Modèles COCOMO : Constructive Cost Model

Depuis sa création en 1989 par BOEHM [Boehm1981], le modèle COCOMO est le modèle le plus populaire dans l'estimation de l'effort de développement logiciel. Il a été construit à partir d'une base de données historique de projets logiciels développés au sein de TRW (société américaine active dans plusieurs domaines, y compris le développement informatique et l'ingénierie des systèmes). Cette base de données comporte 63 projets logiciels développés selon le modèle de cycle de vie en cascade [6] en utilisant des langages allant de l'assembleur au PL/I, et dont la taille varie de 2 000 à 100 000 lignes de codes. Le modèle COCOMO a été construit sur une analyse statistique de la base de données TRW.

COCOMO 81 comporte trois modèles permettant d'affiner l'estimation en avançant dans le projet. Il s'agit des modèles : basique, intermédiaire et détaillé. Le modèle basique correspond aux phases amont alors que les modèles l'intermédiaire et détaillé correspondent aux phases avancées du projet.

Les modèles COCOMO 81 reposent sur l'équation :

$$\widetilde{E} = a(KLOC)^b \times FAE \tag{1.2.3}$$

où $\widetilde{E}$ est l'effort estimé en mois-homme, KLOC est la taille du projet logiciel estimée en kilo (mille) lignes de code source. Les paramètres a et b sont des constantes qui dépendent de la complexité du logiciel (organique, semi-détaché ou intégré) et peuvent être déterminés à partir d'une table (voir annexe A). FAE est le facteur d'ajustement de l'effort, il est ajusté selon le modèle COCOMO

5. La distribution de Rayleigh est une loi de probabilité continue qui représente la norme d'un vecteur gaussien bidimensionnel dont les coordonnées sont indépendantes, centrées et de même variance.

6. Le cycle de vie en cascade (ou cycle de vie séquentiel) est un modèle de développement classique qui se base sur la succession des phases du projet.

81 utilisé comme suit :

$$FAE = \begin{cases} 1 & \textit{(modèle basique)} \\ \prod_{i=1}^{15} C_i & \textit{(modèle intermédiaire et détaillé)} \end{cases} \tag{1.2.4}$$

Où C_i représente la valeur du facteur d'effort i qui peut être déterminée à partir d'une table (voir annexe A).

Afin d'améliorer le modèle COCOMO et le rendre compatible avec les évolutions technologiques opérées dans ce secteur depuis les années 90, une autre version appelée COCOMO II a été développée [Boehm2000]. Ce dernier modèle adapte le COCOMO basique aux nouvelles pratiques de développement logiciel telles que les processus de développement itératif et rapide, ainsi que les approches de réutilisation des composants.

1.2.3.3 Régression Linéaire Simple (RLS)

Les premiers modèles d'estimation algorithmiques sont formulés sur l'hypothèse (1) que la taille du projet est le facteur d'effort principal et unique et (2) que la relation entre l'effort et la taille est linéaire [Fiore+1996 ; Galorath+2006 ; Sadiq2013]. Ces modèles sont également appelés "modèles basés sur la taille" et s'expriment ainsi :

$$\widetilde{E} = \beta_0 + \beta_1 \, Taille \tag{1.2.5}$$

où β_0, β_1 sont les paramètres (coefficients) du modèle d'estimation. Un exemple de ces modèles est le modèle basé sur les cas d'utilisation [Fiore+1996].

Comme leur nom l'indique, les modèles d'estimation par régression linéaire simple sont peu complexes et leur élaboration ne nécessite pas une grande base de données (seulement deux paramètres inconnus à déterminer). Ils peuvent être appliqués très tôt dans le cycle de vie du projet car ils ne s'appuient que sur la taille du projet qui peut être estimée à ce stade. Cependant, ils négligent et ne prennent pas en compte l'influence des autres facteurs qui sont aussi déterminants de l'effort, tels que la productivité et la complexité du projet [Nassif+2011].

1.2.3.4 Régression Linéaire Multiple (RLM)

Le modèle d'estimation d'effort par régression linéaire multiple est une extension à plusieurs dimensions du modèle de régression linéaire. Il est généralement définit par l'équation :

$$\widetilde{E} = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_e X_e \tag{1.2.6}$$

où $\beta_0, \beta_1, \cdots, \beta_e$ sont les coefficients de régression et $X_1, \cdots X_e$ représentent les facteurs d'effort. Ils peuvent être déterminés par plusieurs méthodes appelées méthodes d'estimation de paramètres, telles que la méthode des moindres carrés.

Les modèles d'estimation d'effort RLM se caractérisent par la facilité de leur construction, la simplicité de leur utilisation et leur forme interprétable (les coefficients représentent explicitement l'importance de chaque facteur d'effort). Cependant, l'utilisation de cette technique de modélisation nécessite la vérification de quelques hypothèses, notamment : (a) la relation entre l'effort et les facteurs d'effort est linéaire, (b) l'erreur est une variable indépendante et suit une distribution normale avec une variance constante et une moyenne nulle et (c) l'effort suit une distribution normale [Fedotova+2013].

1.2.4 Modèles d'estimation non paramétriques

Les modèles non paramétriques s'appuient sur moins d'hypothèses concernant la distribution des données et supposent que la forme de la fonction n'est pas définie à priori. Dans ces modèles, le nombre de paramètres est flexible et évolue pour s'accorder avec la complexité des données [Brunet2010]. En effet, ces modèles ont, en plus des coefficients, des hyper-paramètres qui sont des variables contrôlant la configuration des modèles. En effet, le nombre de coefficients est

fonction des hyper-paramètres. La majorité des modèles non paramétriques appartiennent aux algorithmes d'apprentissage automatique, comme par exemple les réseaux de neurones (RNA), les arbres de décision, les forêts d'arbres décisionnels (FAD) et la méthode des k plus poches voisins (k-PPV). Par exemple, dans le cas de RNA, un modèle d'estimation s'appuyant sur cette technique peut être caractérisé par trois hyper-paramètres qui sont le nombre de couches cachées, le nombre de neurones dans chaque couche cachée et les fonctions de transfert utilisées.

1.2.4.1 Réseaux de Neurones Artificiels (RNA)

Les réseaux de neurones sont apparus en 1957 avec le perceptron développé par Rosenblatt, ils ont ensuite connu un grand essor dans les années 1980 grâce au physicien Hopfield qui a présenté clairement, dans son fameux article [Hopfield1982], la théorie de fonctionnement et la structure des réseaux de neurones. Les réseaux de neurones s'appuient sur l'architecture des réseaux neuronaux biologiques constituant le cerveau humain afin de recréer son comportement intelligent. Un réseau de neurones est composé de neurones et synapses et se présente comme un ensemble de nœuds (neurones) connectés entre eux. La figure 1.3 présente la structure d'un neurone artificiel.

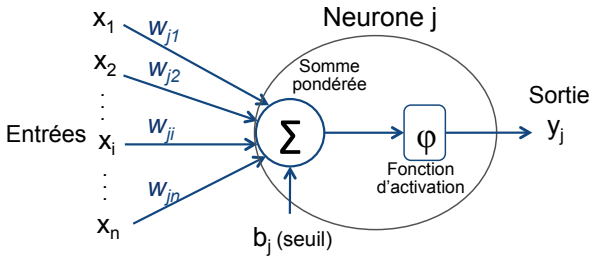


FIGURE 1.3 – La structure d'un neurone artificiel

Chaque neurone j du réseau est un élément processeur, il est aussi définit comme "une fonction non linéaire, paramétrée, à valeurs bornées" [Dreyfus+2011]. Un neurone reçoit des valeurs en entrée x_i associées à des poids w_{ji} représentant l'importance de ces entrées. Il renvoie une valeur unique comme sortie, qui peut être envoyée à plusieurs neurones en aval. Une fonction de combinaison calcule le potentiel du neurone qui est la somme pondérée des entrées et leurs poids à la quelle se rajoute le seuil b_j : $v_j = b_j + \sum x_i w_{ji}$. Une seconde fonction φ appelée fonction d'activation ou fonction de transfert est appliquée à ce potentiel pour générer la valeur en sortie y_j. Il existe plusieurs types de fonction de transfert, la plus utilisée est la fonction sigmoïde avec ses deux formes logistiques (equation 1.2.7) et tangentielle (equation 1.2.8).

$$s(x) = \frac{1}{1+e^{-x}} \tag{1.2.7}$$

$$tanh(x) = \frac{e^{-x}-1}{e^{-x}+1} \tag{1.2.8}$$

L'association et la connexion de plusieurs neurones selon une architecture donnée permet de construire un réseau de neurones. Le type de réseau de neurones le plus courant et le plus simple est le perceptron multi-couche. Son architecture ou structure est présentée dans la figure 1.4. Le perceptron multi-couche est composé d'une couche d'entrée, une couche de sortie et des couches cachées. Sa structure est caractérisée par trois hyper-paramètres : le nombre de couches cachées, le nombre de neurones dans chaque couche cachée et la forme des connexions entre les nœuds (fonctions de transfert utilisées).

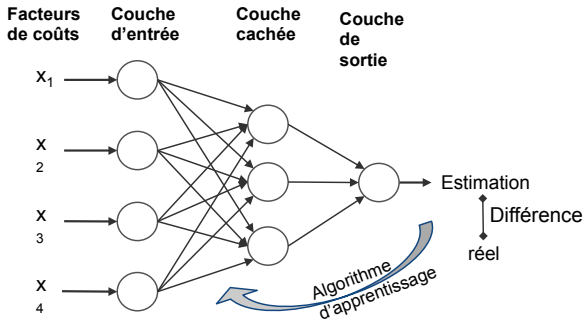


FIGURE 1.4 – La structure d'un réseau de neurones artificiel

Avant de pouvoir utiliser un réseau de neurone dans l'estimation d'efforts de nouveaux projets logiciel, il faut tout d'abord déterminer sa structure optimale et ensuite l'entrainer sur une base de données de projets réalisés. Cette dernière phase s'appelle la phase d'apprentissage. Elle permet d'ajuster les poids des connexions entre les nœuds du réseau de neurones. Cette phase utilise un algorithme qui compare la valeur produite par le réseau de neurone avec le valeur réelle pour chaque individu de la base de données, il ajuste ensuite les poids jusqu'à l'obtention d'un comportement proche du comportement réel. Les algorithmes d'apprentissage les plus courants sont : l'algorithme de rétro-propagation du gradient, l'algorithme de propagation rapide où encore les algorithmes génétiques [Tuffery2010].

Dans la littérature, plusieurs études telles que [Dave+2012 ; Idri+2010 ; Kaushik+2012 ; Reddy+2010], ont proposé des modèles qui se basent sur les réseaux de neurones pour l'estimation de l'effort de développement logiciel. Ces études ont montré que ce type de modèle d'estimation donne de bons résultats. Ils ont souligné l'efficience des réseaux de neurones dans la modélisation des relations et comportements complexes et non linéaires.

1.2.4.2 Forêts d'Arbres Décisionnels (FAD)

La technique des forêts d'arbres décisionnels où des forêts aléatoires a été formellement introduite par BREIMAN en 2001 [Breiman2001]. C'est une technique d'apprentissage automatique utilisée pour résoudre les problèmes de classification et de régression. Une forêt d'arbres décisionnels est un ensemble d'arbres de décision qui sont construits à partir de sous-ensembles aléatoires de la base de données en utilisant la technique de "bagging" [7].

Les arbres de décision sont des outils d'aide à la décision proposés par BREIMAN et al. en 1984 [Breiman+1984]. Un arbre de décision est composé de nœuds et de branches (voir exemple dans figure 1.5). Chaque nœud correspond à un test qui peut être appliqué sur un ou plusieurs attributs décrivant une instance. Dans le cas le plus fréquent, un test correspond à un seul attribut. Les branches représentent les réponses possibles aux tests des nœuds. Le nœud du sommet est appelé racine, il est situé au premier niveau de l'arbre. Les nœuds terminaux sont appelés feuilles, ils représentent les résultats de l'arbre de décision qui peuvent être des classes ou des valeurs selon la nature du problème. Dans le domaine de l'estimation de l'effort de développement logiciel, il s'agit

7. En statistique, le terme "bagging" où "Bootstrap agrégeant" désigne un méta-algorithme utilisé afin d'améliorer la stabilité et la précision d'un modèle. Appliqué sur un modèle d'arbre de décision, il permet de générer un ensemble d'arbre de décision qui forme une forêt d'arbres décisionnels. Il consiste à générer, à partir de la base de données étudiée, B sous-ensembles appelés "bootstrap" de même dimension que la base. Cela est effectué en utilisant un échantillonnage aléatoire avec remplacement. Ensuite, un arbre de décision est généré à partir de chaque échantillon et B arbres de décision sont construits et forme un forêt. L'estimation de l'effort d'un nouveau projet est obtenue en faisant la moyenne des B estimations produites par les arbres de décision composant la forêt.

souvent d'un problème de régression. La figure 1.5 illustre un arbre de décision pour l'estimation de l'effort construit par [Nassif+2013] à partir de la base de données ISBSG [8].

Dans cet arbre de décision, la racine présente la taille fonctionnelle (T) du projet logiciel en Point de Fonction (PF). Si la taille est inférieure ou égale 493.5 PF alors l'effort estimé est égal à 2306.5 mois-hommes. Sinon, il faut passer à l'autre nœud, si le nombre de requêtes (NR) est inférieur ou égal à 203 alors l'effort est estimé à 3025.6 mois-hommes, sinon il est estimé à 4401.5 mois-hommes.

Plusieurs travaux de littérature ont utilisé les arbres de décision pour proposer des modèles d'estimation d'effort notamment [Andreou+2008 ; Elyassami+2013].

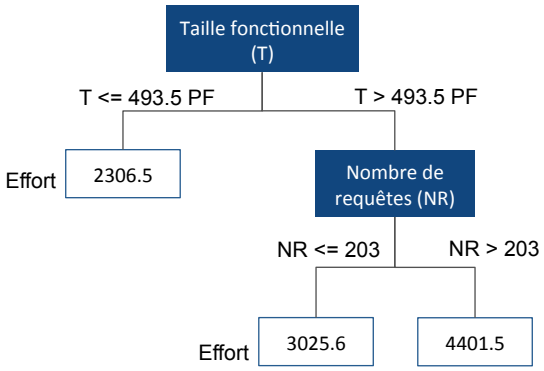


FIGURE 1.5 – Arbre de décision généré par [Nassif+2013] à partir de la base ISBSG

Les forêts d'arbre de décision ne sont pas encore bien exploitées dans le domaine de projet logiciels, peu de travaux de recherche les ont récemment implémentées. Seuls [Shashank Mouli Satapathy2014 ; Suma+2014] les ont récemment utilisées et ont montré qu'elles ont une performance compétitive.

1.2.4.3 k-Plus Proches Voisins (k-PPV)

La technique des k-plus proches voisins (k-PPV) est une technique d'apprentissage supervisé. C'est une des plus simples et fondamentales techniques utilisées pour les problèmes de classification et de régression. k-PPV est un type d'apprentissage à base d'instance dans lequel aucun comportement n'est appris de la base de données et aucun modèle explicite n'est construit. L'apprentissage dans le cas de k-PPV consiste simplement à mémoriser la base de données étudiée. L'estimation de la sortie d'une nouvelle instance est faite en se basant sur les sorties des instances similaires de la base de données [Mitchell1997]. Plus concrètement, dans le contexte de l'estimation de l'effort de développement logiciel et pour l'estimation de l'effort d'un nouveau projet P_n, k-PPV consiste tout d'abord à chercher dans la base de données les k projets les plus similaires à P_n. La similarité peut être mesurée en utilisant des métriques telles que la distance euclidienne ou manhattan. Une fois les k plus proches projets identifiés, leurs efforts sont utilisés pour calculer l'estimation de l'effort de P_n. Cette dernière est obtenue en faisant la moyenne des efforts des k projets trouvés, pondérée par les inverses des distances associées.

La figure 1.6 montre un exemple de l'application de k-PPV (avec k=4) pour la détermination des k plus proches voisins. La similarité dans cet exemple est mesurée en fonction de la taille du projet et la taille de l'équipe.

8. ISBSG est une base de données collectées par le groupe ISBSG (International Software Benchmarking Standards Group). C'est une base de données multi-organisationnelle qui couvre différents entreprises de plus de 25 pays, différent types d'application et plateformes, etc.

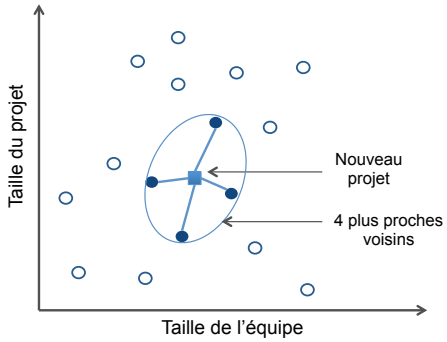


FIGURE 1.6 – Exemple de k-PPV (avec k=4)

1.2.5 Autres classifications de modèles d'estimation

De manière complémentaire à la classification des modèles d'estimation en modèles paramétriques et non paramétriques, nous proposons une autre façon de les regrouper en deux catégories :

— Modèles d'estimation basés sur la théorie : sont généralement des modèles d'estimation fixe et prêts à l'emploi.

— Modèles d'estimation basés sur les données [Hird+2015] : sont des modèles d'estimation à calibrer sur des bases de données de projets réalisés spécifiques aux utilisateurs.

1.2.5.1 Modèles d'estimation basés sur la théorie

Les modèles d'estimation d'effort basés sur la théorie sont dérivés d'hypothèses et de considérations théoriques qui caractérisent certains aspects du processus de développement logiciel. En effet, la théorie est utilisée pour expliquer et décrire, de différentes manières, les relations cause-effet entre l'effort, les objectifs du produit à développer, les atouts de l'entreprise et les contraintes du projet [Fairley1992 ; Hannay+2007].

Généralement, ces modèles sont des "modèles fixes" et "prêts à l'emploi". Ils sont proposés sous une forme générale et prédéfinie, c'est à dire calibrés sur des bases de données générales. De ce fait, ils sont directement utilisables pour l'estimation des efforts de nouveaux projets sans nécessité d'une préparation préalable. Toutefois, il est possible de les ajuster aux données d'un environnement particulier afin de les rendre spécifiques aux projets de cet environnement. Les modèles les plus populaires de cette catégorie sont SLIM et COCOMO.

Les modèles d'estimation basés sur la théorie sont générés et calibrés sur des bases de données qui peuvent être anciennes et obsolètes. L'ensemble des facteurs d'effort utilisé est figé et spécifique aux bases de données d'origine. Par exemple, pour le modèle COCOMO l'utilisateur n'a pas le possibilité d'utiliser une taille fonctionnelle évaluée en PF, et ne peut pas supprimer ou ajouter d'autre facteurs d'ajustement. Cela implique que les nouveaux projets à estimer puissent être décrits conformément à cet ensemble de facteurs. Dans le même exemple de COCOMO, l'utilisateur doit mesurer la taille du projet en SLOC pour pouvoir estimer l'effort et n'a pas la possibilité d'utiliser une autre métrique. Dans le cas contraire, à savoir les nouveaux contextes de projets, ces modèles sont inutilisables et inappropriés.

1.2.5.2 Modèles d'estimation basés sur les données

Les modèles d'estimation basés sur les données s'appuient essentiellement sur une base de données collectées sur les projets réalisés et spécifique à l'utilisateur. Ce mécanisme de construction de modèles à partir d'une base de données est aussi appelé "analyse de régression". Contrairement aux modèles d'estimation d'effort basés sur la théorie, les modèles de cette catégorie sont spécifiques et personnalisables aux contextes et environnements des entreprises. Ils peuvent

également être ajustés et adaptés à différentes phases du cycle de vie. Ils se caractérisent alors par leur flexibilité.

Cette catégorie comprend plusieurs types de modèles tels que RLM, RNA, FAD et k-PPV, les algorithmes génétiques, et les arbres de décision.

La table 1.1 synthétise et croise les différents modèles d'estimation existants selon les deux types de classements : basés sur la théorie/basés sur les données et paramétriques/non paramétriques.

TABLE 1.1 – Classification de modèles d'estimation

	Modèles basés sur la théorie	Modèles basés sur les données
Paramétriques	COCOMO SLIM	RLS RLM
Non paramétriques	———	RNA FAD k-PPV ...

1.2.6 Cartographie des catégories de techniques d'estimation

Vu qu'il existe une pléthore de techniques d'estimation de l'effort de développement, il est bien difficile de désigner quelle est la meilleure technique qui s'appliquerait d'une manière efficace à chaque situation et contexte. En effet, chaque technique d'estimation a ses avantages et ses limitations. Le choix de la technique d'estimation à utiliser dépend du contexte et de l'environnement et des objectifs de l'estimation. La table 1.2 présente une cartographie synthétisant les propriétés générales afin de caractériser et différencier les grandes catégories de techniques d'estimation. Une échelle à quatre niveaux est utilisée pour décrire les réponses d'une technique d'estimation à un critère :

- ++ les techniques d'estimation répondent totalement au critère
- \+ les techniques d'estimation répondent partialement au critère
- − les techniques d'estimation répondent faiblement au critère
- −− les techniques d'estimation ne répondent pas au critère

TABLE 1.2 – Synthèse des propriétés majeures des grandes catégories de techniques d'estimation de l'effort

Critère	Non algorithmiques	Algorithmiques (paramétriques/non paramétriques)	
		Modèles basés sur la théorie	Modèles basés sur les données
Objective	−−	++	++
Répétable	−−	++	++
Ne nécessite pas de base de données	++	−	−−
Ne nécessite pas d'expérience	−−	++	++
Interprétable	−−	++	++
Personnalisable	++	−	++
Flexible	++	−	++
A une bonne puissance prédictive	+	+	++

Comme le montre la table 1.2, la catégorie des modèles d'estimation basés sur les données présente plus d'avantages et correspond le plus au contexte de notre problématique.

Dans la suite de ce travail, nous allons nous intéresser aux modèles d'estimation basés sur les données qu'ils soient paramétriques ou non paramétriques. Après avoir étudié les techniques d'estimation d'effort de projet de développement logiciel existants, nous allons nous intéresser aux incertitudes qui interfèrent sur le processus d'estimation.

1.3 Incertitudes dans le processus d'estimation

Dans le guide pour l'expression de l'incertitude de mesure (GUM), le mot «incertitude» signifie doute [JCGM2008]. Ainsi, l'incertitude d'une estimation d'effort de développement logiciel signifie : doute sur la validité de cette estimation. Ce doute est généralement la conséquence, entre autres, d'une connaissance incomplète, imprécise et/ou limitée des projets de développement logiciel à estimer et de l'instabilité de l'environnement de développement.

Dans cette section, nous identifions tout d'abord les différentes sources d'incertitudes dans le processus d'estimation. Ensuite, nous mettons en évidence l'importance de modéliser les incertitudes et les exprimer dans l'estimation en utilisant une forme probabiliste telle que l'intervalle de prédiction. Nous présentons par la suite les approches existantes permettant d'estimer d'une façon probabiliste l'effort. A la fin, nous présentons les indicateurs utilisés pour évaluer les approches de calcul d'Intervalle de Prédiction (IvP) de l'effort.

1.3.1 Sources d'incertitudes

La figure 1.7 positionne les sources et les différents facteurs d'incertitude correspondant aux deux phases de construction de modèle d'estimation et de l'utilisation de modèle d'estimation construit dans l'estimation de nouveaux projets. Il existe trois sources majeures d'incertitudes dans le processus d'estimation, elles sont relatives à la base de données de projets réalisés, aux modèles ou méthodes d'estimation et à l'évaluation des facteurs d'effort des nouveaux projets à estimer.

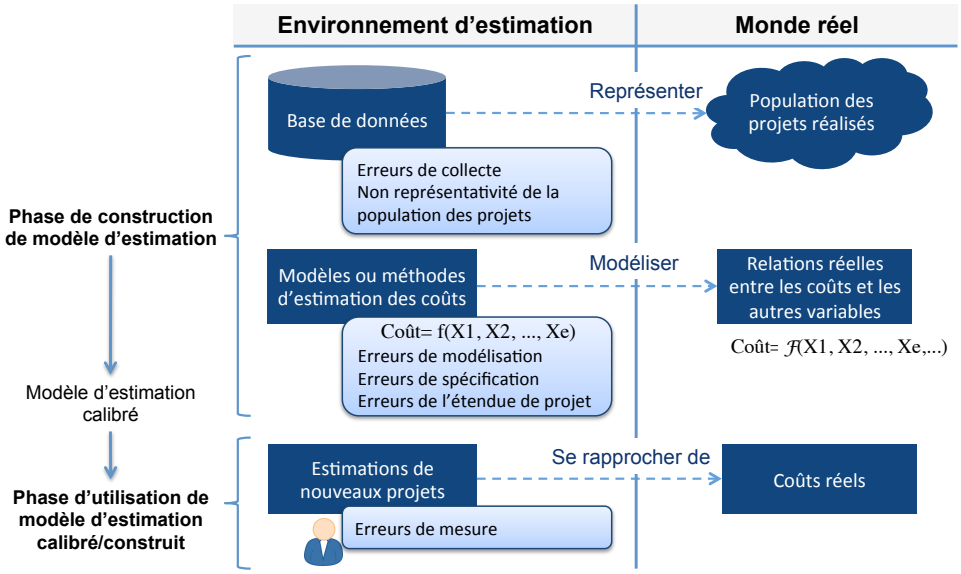


FIGURE 1.7 – Les sources d'incertitudes dans le processus d'estimation

1.3.1.1 Incertitudes liées à la base de données

Les bases de données des projets logiciels réalisés présentent plusieurs sources d'incertitudes qui sont relatives :

— aux erreurs de collecte : bien que la collecte ne concerne que les projets logiciels accomplis, il existe des attributs de projets qui ne peuvent être parfaitement connus et précis, notamment la taille du projet. En effet, ces attributs représentent des caractéristiques abstraites du projet et donc ne sont pas explicites comme le cas du "type du projet" ou de la "taille de l'équipe" par exemple. Par conséquent, les erreurs de mesures et d'estimation de ces attributs induisent des incertitudes dans la base de données. En outre, il existe encore des

difficultés à adopter les bonnes pratiques de collecte de données par des entreprises. Ainsi, il arrive parfois de perdre l'information ou perdre sa qualité et précision suite au délai entre la clôture du projet et la collecte des données.

— à l'échantillon non représentatif de la population : bien souvent, les bases de données collectées par les entreprises sont de petite taille. La collecte des données sur les projets réalisés n'est pas toujours systématiquement réalisée à la fin du projet. Parfois, les données de quelques projets sont partiellement ou pas collectées et dans des cas la collecte ne se fait qu'au niveau de quelques services ou unités. De surcroit, avec l'évolution des projets et de leurs environnements et technologies de développement, la base de données peut devenir relativement ou fortement non homogène. L'expert peut alors se retrouver avec une base de données de petite taille et qui ne représente pas tous les projets de l'entreprise [Adams2002].

1.3.1.2 Incertitudes liées aux modèles d'estimation

Il n'existe pas de modèle parfait. En effet, tous les modèles ne sont que des abstractions et représentations de la réalité. Ils ont comme objectif de produire des réponses se rapprochant le plus d'un comportement réel $\mathscr{F}$ qui peut être complexe. De ce fait, ils constituent des sources d'incertitudes pour l'estimation [Kitchenham+1997].

De plus, il n'existe pas de modèle qui peut intégrer tous les facteurs qui influencent l'effort de développement. Les facteurs qui influencent potentiellement l'effort de développement sont nombreux et quelques uns peuvent être inconnus. En général, les modèles d'estimation se limitent à quelques facteurs d'effort. La non prise en compte explicite des autres facteurs conduit aux erreurs de modélisation.

Les facteurs d'effort sont spécifiés soit en se basant sur l'expérience de l'expert ou en utilisant des procédures et tests statistiques qui reposent sur des hypothèses. Dans les deux cas, la sélection des facteurs d'effort est sujette à l'incertitude et présente des erreurs de spécification.

Un autre facteur d'incertitude est lié à l'utilisation du modèle d'estimation. Bien que les modèles d'estimation d'effort soient élaborés en se basant sur les projets réalisés antérieurement, ils ne peuvent pas être parfaitement précis pour les projets futurs. Ceci est dû au fait que, les projets futurs n'ont pas forcément les mêmes propriétés et caractéristiques que les projets passés. En effet, plusieurs facteurs évoluent avec le temps, notamment les technologies de développement, l'expérience et la compétence des développeurs. Par conséquent, l'utilisation des modèles d'estimation d'effort dans ces contextes induit des erreurs appelées "erreurs de l'entendu de projet". Ces erreurs sont accentuées d'autant plus que le contexte du projet à estimer s'éloigne du contexte des projets de la base de données. C'est le cas par exemple, quand une entreprise utilise un modèle "prêt à l'utilisation" et ancien comme COCOMO sans le calibrer à son contexte et environnement [Kitchenham+1997].

1.3.1.3 Incertitudes liées à l'évaluation des facteurs d'effort du nouveau projet

Les nouveaux projets à estimer se caractérisent par les incertitudes importantes induites par le manque et l'imprécision des informations sur ces projets, surtout en phases amont, ainsi que par la variabilité des données au cours de projets. Lors de l'estimation d'un nouveau projet, l'expert commence par étudier le cahier des charges et les spécifications du projet. Puis, il évalue les facteurs d'effort nécessaires à l'estimation de l'effort. Cette évaluation se fait soit en se basant sur l'expérience et l'intuition de l'expert ou à travers des méthodes de mesure plus au moins objectives. Quoi qu'il en soit, ces facteurs d'effort sont intrinsèquement incertains. Ils ne représentent que des approximations ou estimations des facteurs d'effort réels [JCGM2008]. En général, les procédures de mesure ont des imperfections qui conduisent à des erreurs de mesure. Ces imperfections sont associées, entre autres, à la méthode et aux outils de mesure ainsi qu'aux erreurs d'application de la procédure de mesure par l'expert. Un exemple illustrant les erreurs de mesures est donné par [Kitchenham+1997], pour le cas de mesure de la taille en PF. En fait, la mesure en PF est supposée avoir au moins 12% d'erreur. Ainsi, pour une taille de projet estimée à 1000 PF, les erreurs de mesure conduisent à dire que la taille réelle est entre 880 et 1120 PF.

1.3.2 Nature probabiliste des estimations

Bien souvent, l'effort estimé est exprimé de façon déterministe, c'est à dire sous la forme d'une valeur unique. Cette représentation ne fournit pas d'informations sur l'incertitude de l'estimation. En effet, une estimation ponctuelle ne peut représenter qu'une estimation parmi plusieurs estimation possibles d'effort [Huang+2007]. Les incertitudes qui existent intrinsèquement dans le processus d'estimation font que l'estimation réaliste doit être représentée de façon probabiliste ou possibiliste sous la forme de distribution de possibilité. Du point de vue pratique, la plupart des auteurs indiquent que la représentation des estimations sous la forme de distributions de probabilité ou d'intervalles de prédiction permet de mieux comprendre ces estimations [Angelis+2000 ; Holm2011 ; Kitchenham+1997 ; Kitchenham+2003].

1.3.2.1 Cône d'incertitude

En phases amont les informations disponibles sur un nouveau projet sont souvent abstraites, incomplètes et imprécises. De surcroit, les spécifications du projet peuvent changer au cours du projet ce qui rend nécessaire la ré-estimation ou la révision de l'estimation de l'effort tout au long du cycle de vie du projet. Ce concept a été illustré la première fois par BOEHM [Boehm1981] et a été affiné plus tard par MCCONNELL [McConnell2006]. La figure 1.8 montre l'évolution de la variabilité ou l'erreur des estimations en fonction du temps et du cycle de vie du projet.

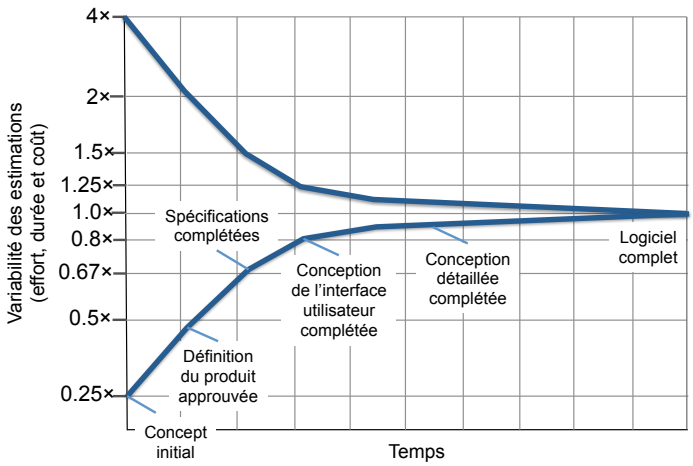


FIGURE 1.8 – Cône d'incertitude de [Boehm1981] affiné par [McConnell2006]

Au début du projet, les estimations sont assujetties à un grand degré d'erreur. En effet, dans la phase de concept initial, les estimations varient entre 25% (×0.25), c.à.d 4 fois moins et 400% (×4), c'est à dire 4 fois plus. La variabilité diminue au fur et à mesure de la progression du projet. Ainsi, à la phase de conception détaillée, elle passe rapidement de ±4 à ±1.25.

1.3.2.2 Estimations sous la forme d'intervalles de prédiction (IvP)

L'incertitude d'une estimation de l'effort d'un projet logiciel peut être exprimée par un "Intervalle de Prédiction" (IvP). Un IvP de l'effort est un intervalle dont les bornes représentent un minimum et un maximum d'effort estimé correspondant à un degré de confiance souhaité. Ce degré de confiance est souvent noté $(1-\alpha)$ où α représente la probabilité que l'IvP ne couvre pas l'effort réel. En effet, l'effort réel devrait appartenir à l'IvP dans $(1-\alpha)$% des cas. Généralement, le degré de confiance peut être différent selon la phase du projet. Par exemple, selon MODER et al. le degré de confiance recommandé pour la phase de planification de projets doit être supérieur ou égal à 90% [Moder+1995].

TABLE 1.3 – Approches de calcul/détermination des IvPs d'effort

Incertitudes \ Approches	non formelles	formelles
Base de données et modèle d'estimation	Jugement d'expert	Distribution de probabilité des performances Bootstrap
Nouveaux projets		Analyse de sensibilité Monte Carlo

Dans quelques travaux, le terme "intervalle de confiance" est aussi utilisé pour désigner l'IvP, notamment dans [Angelis+2000 ; Mittas+2009 ; Papadopoulos+2001]. JORGENSEN et al. souligne que les deux termes "intervalle de confiance" et IvP sont différents [Jorgensen+2003]. En effet, en statistiques, IvP désigne l'incertitude d'une estimation alors que l'intervalle de confiance souvent renvoie à l'incertitude associée aux facteurs d'effort d'un modèle ou d'une méthode d'estimation [Armstrong2002].

1.3.3 Expression des incertitudes dans l'estimation

Différents travaux de littérature proposent de modéliser les incertitudes dans les estimations d'effort à travers des IvPs [Bakır+2010 ; Jorgensen+2002 ; Jorgensen+2003 ; Mittas2011]. Cette section a pour objectif de présenter les différentes approches probabilistes possibles pour calculer/déterminer l'IvP d'effort dans le but de modéliser les incertitudes dans les estimations. Ces approches sont soit non formelles et basées sur l'expérience comme le jugement d'expert, soit formelles et basées sur des modèles. Les approches formelles reposent souvent sur le couplage du modèle d'estimation avec une technique permettant d'intégrer les incertitudes et de les exprimer dans les estimations produites.

Il existe plusieurs techniques permettant d'exprimer les incertitudes provenant de la base de données et du modèle d'estimation. Elles incluent, entre autres, la distribution de probabilité des performances et le bootstrap.

En revanche, peu de techniques permettent de traiter les incertitudes de l'évaluation de nouveaux projets, nous en citons deux techniques : l'analyse de sensibilité et Monte Carlo. Ces techniques se basent sur l'évaluation incertaine des facteurs d'effort du modèle d'estimation à travers des fonctions de densité de probabilité (FDP).

La table 1.3 synthétise l'ensemble des approches de calcul/détermination d'IvP d'effort en fonction des incertitudes qu'elles modélisent.

1.3.3.1 IvPs exprimés par l'expert

Dans cette approche de jugement d'expert, l'expert se base sur son expérience pour estimer l'IvP correspondant à un degré de confiance déterminé. Les experts ont tendance à être optimistes, par conséquent les IvPs des efforts estimés sont très étroits et ne reflètent pas l'incertitude réelle. Entre autres, une analyse effectuée par JORGENSEN et al. sur 18 projets de développement logiciel montre que seulement 64% des IvPs exprimés par les experts contiennent l'effort réel même s'ils correspondent à un degré de confiance élevé de 90% [Jorgensen+2002] .

1.3.3.2 IvPs basés sur la distribution de probabilité des performances

Cette technique utilise les principes statistiques sur les IvPs [Jorgensen+2003] et se base sur les performances des projets réalisés pour dériver le IvP du nouveau projet à estimer. L'estimation des IvPs basés sur la distribution de probabilité consiste en 4 étapes majeures :

1. La sélection de la métrique de performance,
2. La sélection les projets réalisés ayant des incertitudes similaires aux incertitudes du nouveau projet à estimer,

3. L'analyse de la distribution des performances des projets sélectionnés,
4. Le choix du degré de confiance et le calcul du IvP de l'effort estimé.

Cette technique se base sur l'hypothèse que l'expert est capable de sélectionner des projets ayant des incertitudes similaires aux incertitudes du nouveau projet à estimer. La technique est facile à utiliser. Par exemple, afin de calculer l'IvP correspondant à un degré de confiance de 90% en utilisant la distribution empirique, il suffit d'utiliser les percentiles 5% et 95% de cette distribution, respectivement, comme limite inférieure et limite supérieure de l'IvP. Cependant, [Jorgensen+2004b] indique que l'hypothèse sur laquelle se repose la technique n'est pas valide pour tous les contextes et que la connaissance de la distribution des performances des projets similaires reste insuffisante pour obtenir des IvPs toujours consistants.

1.3.3.3 IvPs basés sur le bootstrap

Le "bootstrap" est une autre alternative de calcul des IvPs de l'effort de développement logiciel [Angelis+2000 ; Klas+2011 ; Mittas+2009 ; Port+2008]. C'est une technique statistique de rééchantillonnage qui a été introduite la première fois par [Efron+1994]. Le bootstrap non paramétrique est le type de bootstrap le plus utilisé car il ne fait pas d'hypothèses sur la population étudiée. Dans la suite, le terme bootstrap renvoie au bootstrap non paramétrique.

Le bootstrap permet de calculer l'intervalle de confiance d'un paramètre θ (par exemple la moyenne ou la médiane d'une population inconnue). En effet, à partir d'un échantillon disponible de la population, la procédure de bootstrap consiste à créer aléatoirement un grand nombre d'échantillons B de taille égale à la taille de l'échantillon initial par échantillonnage avec remplacement. Ensuite, chaque échantillon $m = 1, \cdots, B$ est utilisé pour calculer l'estimation $\widetilde{\theta}_m$ du paramètre θ. L'ensemble des $\widetilde{\theta}_m$ composent une distribution empirique. L'IvPs de θ correspondant à un degré de confiance α est donné par $\left[\widetilde{\theta}_{\alpha/2}, \widetilde{\theta}_{1-\alpha/2}\right]$ où $\widetilde{\theta}_{\alpha/2}$ et $\widetilde{\theta}_{1-\alpha/2}$ correspondent respectivement aux percentiles $100(\alpha/2)$ et $100(1-\alpha/2)$ de la distribution de $\widetilde{\theta}_m$.

Dans le domaine de l'estimation de l'effort logiciel, il s'agit du calcul des IvPs de l'effort E [Klas+2011 ; Laqrichi+2015 ; Mittas+2009]. La procédure de bootstrap s'appuie sur la base de données disponible sur les projets logiciels réalisés. Cette technique est très pertinente pour les modèles d'estimation non-paramétriques car elle ne fait pas d'hypothèses sur la distribution des efforts des projets réalisés, telles que la normalité.

1.3.3.4 IvPs basés sur l'analyse de sensibilité

L'analyse de sensibilité permet de modéliser les incertitudes liées à l'évaluation des nouveaux projet. Elle vise à quantifier l'impact de la variation de chaque variable d'entrée sur la réponse du modèle [Saltelli+2009]. La variance de l'effort estimé $v(\widetilde{E})$ est le carré de son incertitude u. Elle est obtenue par propagation de la variance en tenant compte des corrélations possibles entre les facteurs d'effort [Li+2010], selon la formule suivante :

$$v(\widetilde{E}) = u^2(\widetilde{E}) = \sum_{i=1}^{e} \left(\frac{\partial f}{\partial x_i}\right)^2 v^2(x_i) + 2\sum_{i=1}^{e-1}\sum_{j=i+1}^{e} \frac{\partial f}{\partial x_i}\frac{\partial f}{\partial x_j} cov(x_i, x_j) \tag{1.3.1}$$

où v est la variance et cov est la covariance. L'IvP est donnée par $\left[\widetilde{E} - U, \widetilde{E} - U\right]$ où U représente l'incertitude élargie obtenue en multipliant u avec un facteur d'élargissement k typiquement situé dans la plage de 2 à 3. La détermination de k est fonction du degré de confiance souhaité et se base sur des hypothèses sur la distribution des estimation de l'effort probables.

Cette technique ne peut être appliquée que sur les modèles complexes comme les modèles paramétriques. De surcroit, elle se base sur un certain nombre d'hypothèses qui ne sont pas toujours valides, telles que la linéarité du modèle et la normalité de la distribution de l'effort estimé.

Prenons un exemple basé sur le modèle de base de COCOMO pour l'estimation de l'effort des projets de complexité simple, qui s'exprime ainsi : $\widetilde{E} = 2.4 KLOC^{1.05}$ La taille d'un nouveau projet

est estimée à 100 kloc avec une variance de 10 kloc. Ainsi, l'effort est égale à 302 mois-hommes et la variance de l'effort est égale à :

$$v(\widetilde{E}) = u^2(\widetilde{E}) = (2,4 \times 1,5(100)^{0,05})^2 10^2 = 2054 \tag{1.3.2}$$

L'incertitude u est alors égale à 45. L'incertitude élargie est calculée avec un facteur d'élargissement égal à 2 qui correspond à un degré de confiance de 95% : $U = 45 \times 2 = 90$. L'IvP obtenu est alors à $[302-90, 302+90] = [212, 392]$.

1.3.3.5 IvPs basés sur la méthode Monte Carlo

Aussi bien que l'analyse de sensibilité, la méthode Monte Carlo est utilisée pour exprimer les incertitudes liées à l'évaluation des nouveaux projet. Elle vise à propager les distributions des valeurs des facteurs de l'effort au lieu de propager les variances et se base sur l'échantillonnage [Herrador+2005].

Dans cette approche, à chaque facteur d'effort est associé une distribution (normale, triangulaire, etc.) représentative de l'incertitude de l'expert sur la quantification de ce facteur. La méthode Monte Carlo génère un échantillon à partir de chaque distribution représentant un facteur d'effort. Ensuite, l'effort est calculé à partir de l'ensemble des échantillons en utilisant le modèle d'estimation étudié. Ces deux opérations d'échantillonnage et calcul de l'effort sont répétés m fois. Les m efforts sont synthétisés sous la forme d'une distribution de probabilité permettant d'obtenir des intervalles de confiance associés à un degré de confiance déterminé. Monte Carlo permet donc de fournir une approximation numérique de la distribution de l'effort. m doit être assez grand pour permettre une bonne approximation de la distribution réelle de l'effort.

La méthode Monte Carlo est considérée comme la plus effective des simulations numériques de propagation d'incertitudes [LaGrega+2010]. En effet, elle présente entre autre les avantages de ne pas nécessiter de calculs de dérivés et ne pas faire d'hypothèses sur le modèle d'estimation.

1.3.4 Évaluation des approches de détermination d'Intervalles de Prédiction (IvP)

La multitude des modèles d'estimation et des techniques d'expression des incertitudes dans les estimations fait que nombreuses approches d'estimation sous forme d'IvPs peuvent être proposées. Afin de pouvoir les comparer, il existe différents indicateurs permettant d'évaluer leurs performances. Les indicateurs les plus utilisés sont le taux de réussite (*HitRate*), la largeur de l'IvP (*RWidth*) et l'effort réel relatif à l'IvP (*ARPI*).

Le *HitRate* représente la proportion des IvPs contenant l'effort réel. En d'autre terme, cela correspond au pourcentage ou à la probabilité que l'effort réel soit dans l'IvP, ce qui renvoie au degré de confiance. Ainsi, une approche de calcul d'IvPs est performante si son *HitRate* correspond bien au degré de confiance. Le *HitRate* est défini dans l'équation 1.3.3, où $\widetilde{E}_{Min}$ représente l'effort pessimiste correspondant à la limite supérieure de l'IvP et $\widetilde{E}_{Max}$ représente l'effort optimiste correspondant à la limite supérieure de l'IvP.

$$HitRate = \frac{1}{n}\sum_{i=1}^{n} h_i, \quad h_i = \begin{cases} 1, & \widetilde{E}_{Min} \leq E \leq \widetilde{E}_{Max} \\ 0, & E > \widetilde{E}_{Max} \vee E < \widetilde{E}_{Min} \end{cases} \tag{1.3.3}$$

Dans le cas où deux approches ont le même *HitRate*, l'approche produisant les IvPs les plus étroits est considérée plus informative et indique une utilisation plus efficace des informations relatives aux incertitudes. Cette propriété est évaluée à travers le *RWidth* défini dans l'équation 1.3.4 où E_p représente l'effort le plus probable.

$$RWidth = \frac{\widetilde{E}_{Max} - \widetilde{E}_{Min}}{\widetilde{E}_p} \tag{1.3.4}$$

Le dernier indicateur *ARPI* permet d'examiner la distribution de l'effort réel relativement aux IvPs et d'analyser les biais existant dans les IvPs. *ARPI* est la mesure de la distance entre l'effort réel et le point du milieu de l'IvP (IvP_{milieu}), normalisée par la largeur de l'IvP (équation 1.3.5) [Jorgensen+2004a]. L'indicateur *ARPI* donne des valeurs proches de -0.5 et 0.5 quand l'effort réel est proche de la limite inférieure et de la limite supérieure de l'IvP, respectivement. Il est égal à 0 quand l'effort réel correspond au point du milieu de l'IvP. Un *ARPI* qui n'appartient par à l'intervalle [-0.5, 0.5] montre que l'effort réel n'appartient pas à l'IvP de l'effort.

$$ARPI = \frac{E - IvP_{milieu}}{\widetilde{E}_{Max} - \widetilde{E}_{Min}} \tag{1.3.5}$$

1.4 Sélection d'un modèle d'estimation d'effort

Face au nombre important de modèles d'estimation de l'effort, la question qui se pose effectivement est : quel est le "meilleur" modèle d'estimation ?

Le terme "meilleur" dans ce contexte signifie : ayant la meilleure capacité prédictive, comparé aux autres modèles d'estimation candidats. Plusieurs études ont été menées pour répondre à cette question en comparant les modèles d'estimation existants. Généralement, les modèles d'estimation sont évalués par des indicateurs de performance selon une méthode de validation. Avant de présenter ces méthodes d'évaluation et de validation, il est important de définir les concepts décrivant la capacité prédictive de modèles d'estimation.

1.4.1 Exactitude, précision et biais

La capacité prédictive de modèles d'estimation d'effort peut être caractérisée par trois concepts : la précision, le biais et l'exactitude. La plupart des travaux de la littérature utilise le terme "précision" pour désigner également "l'exactitude" ou le "biais". Cependant, il est important de souligner que les trois termes décrivent trois différents aspects.

La précision décrit la variabilité ou la dispersion du modèle d'estimation. Ce concept concerne principalement le mode probabiliste. Dans ce mode, l'estimation d'un projet est exprimée sous la forme d'une plage de valeur, d'une distribution ou d'un IvP. La précision décrit alors la variabilité des estimations possibles d'effort de ce projet, elle peut être utilisée pour quantifier les incertitudes des estimations. En mode déterministe, elle décrit la capacité du modèle d'estimation à produire, pour des projets similaires (ayant l'effort et les facteurs d'effort similaires), des estimations très proches les unes des autres sans qu'elles soient nécessairement proches des valeurs réelles. En d'autres termes, elle décrit la dispersion des estimations de l'effort des projets similaires autour de leurs moyenne (des estimations).

Le biais est l'erreur systématique, il représente la distance entre l'effort réel et l'effort estimé. Il indique aussi la tendance du modèle d'estimation à sur estimer ou à sous estimer.

L'exactitude (en anglais, accuracy), quand à elle, est une fonction de la précision et du biais, elle représente le degré de proximité des estimations aux valeurs réelles. En d'autres termes, elle décrit la dispersion des valeurs estimées autours des valeurs réelles.

La figure 1.9 illustre, dans le cas d'une estimation probabiliste (plage de valeur) d'un projet, les concepts de la précision, du biais et de l'exactitude et les relations qui existent entre eux. Le centre des cercles est l'effort réel et les points sont les efforts estimés. Face à un problème de comparaison de modèles d'estimation d'effort, un aspect peut être priorisé afin de décider sur le meilleur modèle d'estimation ou le modèle d'estimation le plus "performant". Cette priorité est assignée selon les objectifs et le point de vue de l'utilisateur. MIYAZAKI et al. considère que l'exactitude est plus importante que la précision (stabilité) dans le domaine de l'estimation de l'effort de développement logiciel [Miyazaki+1994]. Ce constat se base sur le fait que le domaine logiciel est caractérisé par la présence fréquente de valeurs aberrantes qui influencent fortement la précision.

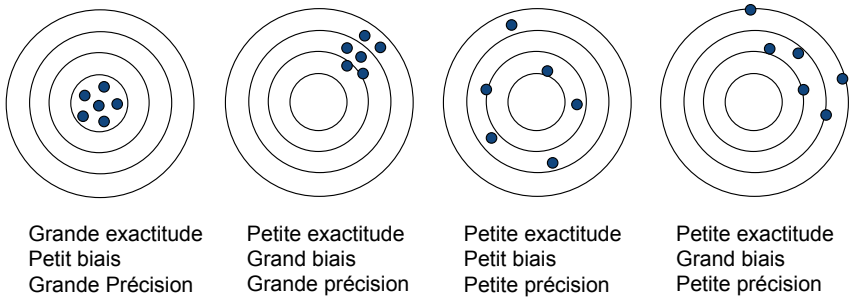


FIGURE 1.9 – Relations exactitude, biais et précision

1.4.2 Évaluation des modèles d'estimation

Pour sélectionner un modèle d'estimation par un ensemble de modèles candidats, il faut être capable de les évaluer tous selon les mêmes critères. Nous allons dans cette section présenter plusieurs critères d'évaluation.

Il s'agit d'évaluer quantitativement la capacité prédictive des modèles d'estimation. Plusieurs métriques ont été utilisées comme indicateurs de performance servant à évaluer les différents aspects de cette capacité. Ci-dessous, sont décrits les indicateurs de performance les plus utilisés dans le domaine de l'estimation de l'effort de développement logiciel.

Pour évaluer une estimation i, [Conte1986] propose d'utiliser la magnitude de l'erreur relative *MRE* défini comme :

$$MRE_i = \frac{|\, E_i - \widetilde{E}_i \,|}{\widetilde{E}_i} \tag{1.4.1}$$

L'évaluation d'un modèle d'estimation requiert souvent l'évaluation de plusieurs estimations correspondantes aux projets de test. La magnitude moyenne de l'erreur relative *MMRE* (Magnitude of the relative Error) est l'indicateur de performance le plus communément utilisé pour ce propos (équation 1.4.2). Selon [Conte1986] un bon modèle d'estimation correspond à un *MMRE* inférieur ou égale à 0.25.

$$MMRE = \frac{1}{n}\sum_{i=1}^{n} MRE_i \tag{1.4.2}$$

Cet indicateur a été critiqué dans différentes études [Foss+2003 ; Kitchenham+1999 ; Myrtveit+2005 ; Shepperd+2012] pour être significativement influencé par les valeurs extrêmes ou aberrantes de *MRE*. De surcroit, il pénalise la sur-estimation plus que la sous-estimation [9]. Une solution alternative permettant d'éviter l'influence des *MRE* extrêmes ou aberrants est l'utilisation de la médiane des *MRE* (*MdMRE*) (équation 1.4.3) plutôt que la moyenne [Jorgensen1995]. Le *MdMRE* est considéré plus représentatif des estimations typiques que la moyenne *MMRE*.

$$MdMRE = Médiane_{i=1}^{n}(MRE_i) \tag{1.4.3}$$

Pred(l) est un autre indicateur de performance assez fréquemment utilisé dans la littérature. Il représente la probabilité qu'un projet ait une erreur inférieure ou égale à l. Il est défini dans l'équation 1.4.4 avec l est le niveau de tolérance et p est le nombre de projets ayant une *MRE* inférieure ou égale à l. Généralement, l prend la valeur 0.25. Selon [Kitchenham+1999], un bon

9. Pour illustrer ceci, nous présentons l'exemple de deux projets A et B : A sur-estimé et B est sous-estimé, mais les deux projets ont le même écart absolu entre l'effort estimé et l'effort réel. Le projet A à un effort estimé de 120 mois-hommes et un effort réel de 100 mois-hommes, alors que B a un effort estimé de 100 et un effort réel de 120. Ainsi, le *MRE* de A est égal à 0.2 alors qu'il est égal à 0.16 pour B. Le *MRE*montre une meilleure performance pour la sur-estimation estimation (projet B) et pénalise la sur-estimation (projet A).

modèle doit avoir un *Pred*(0.25) supérieur ou égal à 0.75.

$$Pred(l) = \frac{p}{n} \tag{1.4.4}$$

Selon des études [Kitchenham+2001 ; Miyazaki+1994 ; Trendowicz2014] *MMRE* et *MdMRE* et *Pred*(l) sont tous utilisés pour mesurer l'exactitude des modèles d'estimation. Le biais et la dispersion peuvent être évalués en utilisant , respectivement, l'erreur moyenne (*AR*) et l'écart quadratique moyen (*RMSE*) (équation 1.4.5).

$$AR = \frac{1}{n}\sum_{i=1}^{n}(E_i - \widetilde{E}_i) \quad RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\widetilde{E}_i - E_i)^2} \tag{1.4.5}$$

Ces indicateurs ne représentent que quelques propriétés caractérisant la distribution des erreurs du modèle d'estimation. Cette représentation a été critiquée par [Kitchenham+2001] qui ont souligné l'importance d'analyser intégralement la distribution des erreurs. Pour cela, ils ont suggéré d'utiliser le diagramme en boîtes ou "boîte à moustaches" afin d'avoir une représentation graphique permettant une analyse complète des comportements des modèles d'estimation d'effort. Le diagramme en boîtes permet de représenter la distribution d'une variable ou d'un ensemble de valeurs qui peut être, par exemple dans notre cas, l'ensemble des erreurs *MRE* obtenu de l'évaluation d'un modèle d'estimation.

La figure 1.10 décrit les composants d'une boite à moustaches des *MRE* d'un modèle d'estimation. Le 1^{er} quartile Q1 (respectivement le 3^{e} quartile Q3) correspond à la plus petite valeur de *MRE* telles qu'au moins 25% (respectivement 75%) des valeurs de *MRE* sont inférieures ou égale à Q1 (respectivement Q3). Les extrémités inférieure et supérieure de la moustache sont déterminées à partir de l'écart inter-quartile (Q3-Q1), elles correspondent à Q1-1,5.(Q3-Q1) et Q3+1,5.(Q3-Q1), respectivement.

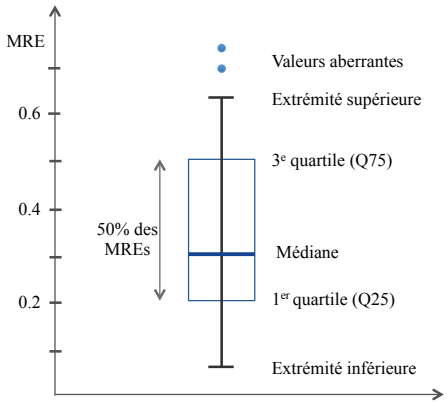


FIGURE 1.10 – Diagramme en boîte à moustaches des *MRE* d'un modèle d'estimation

1.4.3 Validation de modèles d'estimation

La validation d'un modèle d'estimation consiste à en estimer les indicateurs de performance tels que le *MMRE*, *MdMRE* et *Pred*. L'estimation de ces indicateurs s'effectue sur des projets de test qui doivent être indépendants des projets utilisés dans l'élaboration du modèle d'estimation. La stratégie de partition de la base de données en une partie d'apprentissage et une partie de test est défini par des méthodes telles que la méthode "holdout", la validation croisée à "k-blocs" ou la validation croisée "Leave-One-Out".

Nous allons dans les sections suivantes présenter ces différentes techniques de validation.

1.4.3.1 La méthode "holdout"

La méthode "holdout" est la technique de validation croisée le plus simple. Elle consiste à diviser, de manière aléatoire, la base de donnée en deux partie ; une partie dite d'apprentissage (généralement, au minimum 60% de la base de données) et une autre dite de test. Le modèle d'estimation à évaluer est construit sur la partie d'apprentissage et testé sur la partie de test [Refaeilzadeh+2009]. Le test consiste à utiliser le modèle construit pour l'estimation des projets et de dériver les indicateurs de performance à partir des estimations et des valeurs des projets de test. Cette méthode est utilisée quand la taille de la base de données est suffisamment grande. En effet, dans le cas d'une grande base de données, sa division pour réserver une partie au test ne compromettra pas l'apprentissage correct du modèle d'estimation. Dans le cas contraire, les méthodes de validation croisées "K-fold" et "Leave-One-Out" sont préconisées.

1.4.3.2 Validation croisée "K-fold"

La validation croisée "K-fold" ou "K-blocs" (VCK) est la procédure de validation la plus populaire, en particulier dans le domaine de l'estimation logiciel où souvent les bases de données disponibles sont de petite ou moyenne taille.

La procédure de VCK est illustrée dans la figure 1.11. La base de données est divisée en k blocs disjoints de tailles sensiblement égales. Un bloc i est réservé pour le test du modèle et les autres (k-1) blocs sont utilisés comme une base d'apprentissage du modèle. Le modèle est entrainé sur la base d'apprentissage, le modèle construit est appelé "modèle de substitution". Il est ensuite testé sur le bloc de test, puis la performance associée est calculée. Cette opération est réitérée pour tout les blocs i variant entre 1 et k. A la fin de cette boucle de k itérations, la performance du modèle d'estimation est obtenue en moyennant les performances des k modèles de substitution.

Le choix de k (la proportion de la base d'apprentissage par rapport à la base de test) se fait en testant des valeurs, qui sont généralement inférieures à 10, de telle sorte qu'il optimise la fiabilité de l'évaluation et le coût de calcul. Ce dernier est fonction du nombre d'itérations. En général, k est choisi entre 5 et 10 [Arlot+2010].

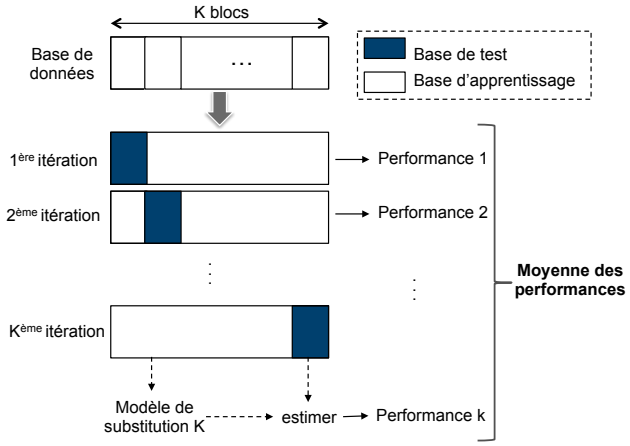


FIGURE 1.11 – Procédure de la validation croisée "k-fold"

1.4.3.3 Validation croisée "Leave-One-Out"

La validation croisée "Leave-One-Out", qui peut être traduite par "validation par exclusion d'un élément", est un cas particulier de VCK où K=N (N étant la taille de la base de données). En effet, à chaque itération, l'apprentissage se fait sur (N-1) projets de la base de données et le modèle de substitution construit est testé sur le seul projet qui reste [Arlot+2010].

1.4.4 Études de comparaison de modèles d'estimation

Plusieurs travaux ainsi que des revues de littérature ont été conduits dans le but de comparer les modèles d'estimation existants.

Dans le cadre d'une récente revue systématique sur les modèles d'estimation basés sur l'analogie (EbA), [Idri+2015] présente les résultats de comparaison de plusieurs travaux de recherche. Les modèles EbA sont comparés avec les modèles de RLM sur différentes bases de données. Les résultats de 38 études montrent qu'il y a un désaccord dans la conclusion sur le meilleur modèle d'estimation, cependant la plupart des études favorisent les RLM.

La même contradiction avait été relevée dans un article plus ancien [Mair+2005]. En effet, parmi les 20 études, sept ont fini par favoriser la RLM, neuf ont appuyé l'EbA et quatre sont restées indifférentes et n'ont pas conclu sur le meilleur modèle d'estimation. Les études de comparaison ont été faite sur différentes bases de données, il semble alors évident d'avoir des résultats contradictoires. Cependant, [Mair+2005] a souligné que parmi les études examinées, certaines sont faites sur la même base de données et pourtant elles montrent des résultats contradictoires.

Dans une autre revue systématique sur les modèles d'estimation basés sur l'apprentissage automatique, [Wen+2012] compare, entre autres, les modèles basés sur les RNA avec les RLM. Similairement à la revue de [Idri+2015], les études de comparaison utilisent différentes bases de données. Parmi 11 études de comparaison, 8 favorisent le RNA et 3 favorisent le RLM. Ces contradictions sont d'autant plus importantes dans les comparaisons effectuées entre d'autres modèles, à savoir les modèles RNA et les modèles d'arbres de décision.

La première conclusion à tirer de ces travaux est qu'il n'existe pas de "meilleur" modèle d'estimation pour tous les contextes et environnements. La deuxième conclusion est que les travaux de comparaison de modèles d'estimation présentent des résultats instables et inconstants [10].

1.4.5 Sources d'inconstance des résultats de comparaison de modèles d'estimation

Plusieurs études ont tenté d'apporter une analyse au problème d'inconstance et d'instabilité des résultats de comparaison de modèles d'estimation [Keung+2012 ; Menzies+2012 ; Mittas+2015a]. Les sources potentielles de cette inconstance sont : la qualité des bases de données, la pertinence des indicateurs de performance et les biais de la méthode de validation.

1.4.5.1 Qualité des bases de données

La performance d'un modèle d'estimation est considérablement influencée par la qualité de la base de données [Azzeh2012 ; Shepperd+2001]. Il n'est donc pas surprenant qu'un modèle d'estimation ait différentes performances sur différentes base de données.

Généralement, des techniques de traitement et de préparation sont appliquées sur les bases de données avant leur utilisation afin d'améliorer leur qualité. Cela inclut par exemple la méthode pour déterminer les facteurs d'effort et la méthode pour traiter les valeurs aberrantes. Cependant, il n'existe pas de façon unique pour procéder à cette préparation. Par conséquent, la performance d'un modèle d'estimation se basant sur une base de données particulière peut aussi varier d'une étude à une autre selon la procédure de préparation utilisée. De surcroit, les études existantes de comparaison de modèles ne fournissent pas assez d'information ni de détails sur la procédure de préparation adoptée. Il est donc difficile de confronter leurs résultats et d'en déduire le meilleur modèle d'estimation pour une base de données particulière.

1.4.5.2 Pertinence des indicateurs de performance

La comparaison de modèles d'estimation dépend également des indicateurs de performance choisis et de leur interprétation [Trendowicz2014]. En effet, il n'existe pas de système d'indicateurs

10. L'inconstance dans ce contexte est définit comme la disposition à se changer d'une étude à une autre.

de performance standard pour l'évaluation des modèles d'estimation. Différents systèmes d'indicateurs de performance sont utilisés dans les comparaisons selon la vision de l'auteur. De surcroit, l'ensemble des indicateurs de performance utilisé par quelques auteurs ne reflètent pas tous les aspects importants du comportement des modèles. D'autre part, les indicateurs de performance les plus communément utilisés (*MMRE* par exemple) sont biaisés étant donné qu'ils se basent sur les *MRE*s. Ces derniers décroient en fonction de l'effort et pénalisent les sur-estimations plus lourdement que les sous-estimations [Foss+2003 ; Kitchenham+2001].

1.4.5.3 Biais des méthodes de validation

Une autre source d'inconstance des résultats de comparaison est relative aux biais de méthodes utilisées pour la validation des modèles d'estimation d'effort. Les premiers travaux sur la comparaison de modèles d'estimation ont utilisé la méthode "holdout". Cette méthode a l'avantage d'être simple mais elle présente des biais considérables. En effet, la stratégie de répartition de la base de donnée utilisée peut induire à la création d'une base de test qui n'est pas représentative de la la base de données globale, ce qui conduit à des évaluations erronées d'indicateurs de performance.

Aujourd'hui, les chercheurs optent plutôt pour les méthodes de validation croisée et surtout la méthode VCK. Cette méthode a été récemment examinée par des chercheurs dans les domaines de bio- et chimio- informatiques [Reunanen2003 ; Varma+2006]. Il apparait que la VCK est fiable pour l'évaluation des modèles d'estimation paramétriques. Elle est cependant biaisée pour le cas des modèles d'estimation non paramétriques étant donnée que la détermination des hyper-paramètres et l'évaluation de modèles d'estimation sont indépendantes. En effet, les hyper-paramètres sont déterminés de telle sorte qu'ils optimisent la performance du modèle d'estimation. Cette performance optimale est ensuite rapportée comme étant la performance réelle du modèle d'estimation. Ceci conduit à une évaluation optimiste et biaisée de la performance réelle du modèle d'estimation.

Conclusion

Nous avons procédé dans ce chapitre à une analyse des différentes techniques d'estimation existantes. La conclusion qui en découle est que pour avoir des estimations d'effort plus proches des efforts réels et qui sont adaptées au contexte de l'organisation, l'expert doit privilégier les modèles d'estimation basés sur les données historiques propres à l'organisation.

Face aux différents modèles existants d'estimation basés sur les données, il est important pour l'organisation de pouvoir sélectionner le modèle plus adéquat pour son contexte. Malheureusement, il n'existe pas de "meilleur" modèle d'estimation pour tous les contextes et environnements. Par conséquent, la comparaison de modèles d'estimation doit être effectuée d'une façon indépendante sur chaque base de données représentant l'environnement propre à l'organisation.

Nous avons étudié les travaux existants sur la comparaison de modèles d'estimation. Sur une même base de données, ces travaux montrent des résultats contradictoires et inconstants. Il n'existe pas de procédure ou de démarche standards supportant ces comparaisons. Plusieurs sources peuvent biaiser les résultats de comparaison, à savoir la procédure de préparation de la base de données, les indicateurs de performance et la méthode de validation utilisés. Ainsi, il est important de proposer une approche d'évaluation et de comparaison fiable de modèles d'estimation afin de sélectionner le modèle le plus adéquat pour une base de données.

Nous avons identifié trois types d'incertitudes interférant sur le processus d'estimation et qui sont relatifs : à la base de données, aux modèles d'estimation et aux nouveaux projets.

Les modèles d'estimation et les bases de données des organisations, étant entachés d'incertitudes, il est importance prendre en compte ces incertitudes dans l'approche de sélection d'un modèle d'estimation. Nous proposons d'utiliser un critère de sélection qui quantifie les incertitudes des modèles d'estimation sur une base de données spécifique.

Très souvent, une estimation est exprimée de façon déterministe sous la forme d'une valeur unique, ce qui n'est pas une façon représentative de la réalité. En effet, l'environnement d'estimation est intrinsèquement incertain. Pour que l'estimation soit réaliste, nous avons souligné l'importance de modéliser et décrire toutes les incertitudes principales identifiées dans les estimations de l'effort de développement. Une représentation plus réaliste des estimations peut être obtenue à travers des distributions de probabilité ou des intervalles de prédiction plutôt que des valeurs uniques. Plusieurs techniques statistiques peuvent être utilisées pour représenter les incertitudes et calculer les intervalles de prédiction de l'effort. Pour notre contexte, nous préconisons l'utilisation de la technique bootstrap pour modéliser les incertitudes des modèles d'estimation et de la technique Monte Carlo pour traiter les incertitudes relatives aux nouveaux projets. Ces techniques sont opérationnelles et présentent l'avantage de ne pas s'appuyer sur des hypothèses sur la base de données à laquelle elles seront appliquées.

CHAPITRE

2

Sélection de modèle d'estimation d'effort en prenant en compte les incertitudes

Introduction

Il est important pour une organisation de pouvoir estimer d'une façon réaliste l'effort de ses nouveaux projets de développement logiciel. En raison de leur flexibilité et adaptabilité au contexte et à l'environnement de l'organisation, les modèles d'estimation basés sur les données de l'organisation sont les plus adaptés pour répondre à ses objectifs.

Face à la multitude de modèles d'estimation d'effort existants de ce type, la question se pose de (1) savoir lequel de ces modèle d'estimation est le plus adapté à la situation (organisation, culture,

expérience, pratiques, etc.) et (2) comment ensuite le calibrer pour le faire correspondre au mieux à la situation et ainsi construire un système d'estimation performant.

Dans cette optique, le présent chapitre vise à proposer une approche structurée permettant d'assister les organisations dans la sélection du modèle d'estimation le plus adapté à leurs besoins, contexte et environnement. Cette approche permettra la comparaison fiable de plusieurs modèles candidats et la sélection d'un modèle d'estimation selon plusieurs critères. Il apportera des réponses aux verrous scientifiques suivants :

— Comment évaluer d'une façon fiable la performance prédictive d'un modèle d'estimation sur la base de données de l'organisation. En d'autres termes, comment évaluer la capacité d'un modèle d'estimation à produire des estimations d'efforts proches des efforts réels. L'évaluation fiable doit reposer sur des techniques de validation qui permettent de réduire les biais ainsi que sur des critères de sélection pertinents.

— Comment évaluer et prendre en compte les incertitudes d'un modèle d'estimation sur une base de données lors de la comparaison et sélection de modèle. Il s'agit de deux types d'incertitudes : celles relatives au modèle d'estimation et celles relatives à la base de données de l'organisation. Ces incertitudes doivent être conjointement évaluées avec la performance prédictive.

2.1 Formulation du problème de sélection de modèle d'estimation

Notre approche de construction d'un système d'estimation d'effort [1] s'appuie sur une base de données des projets réalisés. Il s'agit d'une base de données spécifique à une organisation. La base de données peut être représentée par une matrice S (eq 2.1.2) de dimension $N \times L$, avec N représente le nombre des projets collectés et L le nombre de variables ou attributs décrivant ces projets.

Chaque projet i de la base de données est représenté par un vecteur R_i composé de valeurs d'attributs a_{ij} et de l'effort réel E_i avec $1 \leq i \leq N$.

$$R_i = (a_{i1}, a_{i2}, \cdots, a_{iL}, E_i), 1 \leq i \leq N \tag{2.1.1}$$

L'ensemble des vecteurs d'attributs caractérisant ces projets est représenté par $A_j (1 \leq j \leq L)$ et noté aussi $\mathscr{A}$.

L'ensemble des efforts réels des projets de la base de données est représenté par $E = \{E_i, 1 \leq i \leq N\}$ où E_i représente l'effort du projet i.

$$S = \begin{pmatrix} Projets \\ R_1 \\ R_2 \\ \vdots \\ R_N \end{pmatrix} = \begin{pmatrix} A_1 & A_2 & \cdots & A_L & E \\ a_{11} & a_{12} & \cdots & a_{1L} & E_1 \\ a_{21} & a_{22} & \ddots & a_{2L} & E_2 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NL} & E_N \end{pmatrix} \tag{2.1.2}$$

Soit $\mathscr{M} = \{M_1, M_2, \cdots, M_c\}$ l'ensemble des modèles d'estimation candidats. Ces modèles d'estimation sont évalués en utilisant des indicateurs de performance pour pouvoir les comparer et sélectionner le meilleur modèle d'estimation.

L'évaluation des critères de sélection repose sur des méthodes de validation, notamment la VCK qui est la plus utilisée jusqu'à présent. Ces méthodes peuvent être considérées comme des estimateurs

1. Un système d'estimation d'effort est un modèle ou ensemble de modèles d'estimation calibrés sur la base de données étudiée. Il est prêt à être utilisé pour l'estimation d'effort de nouveaux projets.

des performances inconnues de modèles d'estimation. L'erreur induite par une méthode de validation V dans l'évaluation de la performance peut être décomposée principalement en biais et variance et peut être exprimée comme suit :

$$Erreur_V(\widetilde{P}(M,S)) = Biais(\widetilde{P}(M,S)) + Variance(\widetilde{P}(M,S)) \tag{2.1.3}$$

où $\widetilde{P}(M,S)$ est la performance du modèle M estimée par la méthode de validation V sur la base de données S. Les biais d'une méthode de validation sont la différence entre la performance réel et la performance estimée/évaluée du modèle d'estimation par cette méthode de validation. La variance reflète la variabilité de la méthode de validation autour de la moyenne de la performance estimée du modèle d'estimation. Cette variance dépend du plan d'échantillonnage utilisé dans la méthode de validation. Dans le cas de la VCK, la variance diminue en augmentant le nombre k de blocs. Cependant, les biais de la VCK restent importants et non négligeables. L'objectif de notre travail de recherche est alors de réduire ces biais pour une comparaison plus fiable de modèles d'estimation.

La performance d'un modèle d'estimation peut être représentée par un ensemble d'indicateur de performance qu'on note $P = (p_1, p_2, ..., p_q)$. Ces indicateurs doivent inclure les mesures des différents aspects de performance prédictive de modèle (exactitude, précision, biais) ainsi que la mesure des incertitudes de modèle d'estimation calibré sur la base de données. Notre approche vise à proposer une procédure d'évaluation de modèles d'estimation et des indicateurs de performance pertinents.

Le problème de sélection d'un modèle d'estimation d'effort s'exprime alors comme un problème d'optimisation multi-objectifs : chercher le modèle d'estimation permettant d'optimiser la performance d'estimation. Notre approche vise à aider et orienter l'utilisateur (expert, chef de projet, etc.) à la sélection d'un modèle d'estimation adéquat pour son organisation et qui optimise les performances.

2.2 Approche de sélection de modèle d'estimation

L'approche que nous proposons dans notre étude repose sur la base de données de l'organisation et permet de comparer un ensemble de modèles d'estimation choisi par l'utilisateur (chef de projet, expert d'estimation, etc.). Comme présentée dans la figure 2.1, notre approche comporte

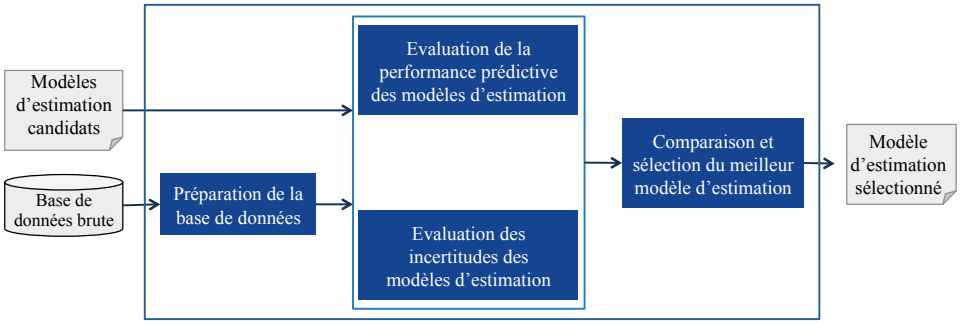


FIGURE 2.1 – Approche de sélection de modèle d'estimation

quatre étapes majeures :

— La préparation de la base de données : Pour construire un "bon" modèle d'estimation, il est nécessaire de disposer d'une base de données complètement renseignée (sans information manquantes) et de bonne qualité. Généralement, les bases de données des organisations contiennent des erreurs, des bruits et des données manquantes ou erronées. Une préparation préliminaire est souvent nécessaire pour pouvoir supprimer de la base ces erreurs.

— L'évaluation de la performance prédictive des modèles d'estimation sur la base de données étudiée : l'objectif est d'évaluer d'une façon fiable (non biaisée) la performance de généralisation (proximité des efforts estimés aux efforts réels) des modèles d'estimation en utilisant différents indicateurs.

— L'évaluation des incertitudes des modèles d'estimation sur la base de données étudiée : elle vise à permettre de prendre en compte, dans la comparaison de modèles d'estimation, les incertitudes aussi bien que la performance prédictive. Ceci permettra d'assurer la sélection d'un modèle d'estimation qui possède à la fois une bonne performance prédictive et une bonne précision.

— La comparaison et la sélection du meilleur modèle d'estimation pour la base de données étudiée : elle repose sur des critères de sélection évaluant les différents aspects de la performance prédictive ainsi que les incertitudes des modèles d'estimation. Dans cette activité nous proposons des orientations pour guider l'utilisateur dans la sélection du meilleur modèle selon ses objectifs et besoins.

Nous détaillons dans la suite chacune de ces activités en proposant des solutions aux verrous rencontrés.

2.3 Préparation de la base de données

Les données sur les projets logiciels réalisés sont collectées au niveau des différents services de l'organisation qui participent aux projets ou par les chefs de projet.

Les bases de données historiques sont des représentations imparfaites du monde réel. En effet, elles présentent naturellement des erreurs de mesure, des données erronées, des valeurs aberrantes et des bruits (voir section 1.3.1.1). L'utilisation de la base de données dans son état brut peut conduire à des résultats incohérents et biaisés (Garbage In Garbage Out). Une base de données de bonne qualité, c.à.d. complète, homogène et cohérente est donc essentielle pour une modélisation pertinente et performante.

La préparation de la base de données, connue aussi sous l'expression d'exploration de données ou encore data mining est un moyen permettant une analyse de la base de données brute et une extraction des informations utiles pour la construction d'un bon système d'estimation d'effort.

Jusqu'à présent, il n'existe pas de procédure standard et d'outil automatique pour effectuer cette tâche. De nombreuses techniques statistiques peuvent être utilisées pour effectuer différents types de traitement. Nous proposons une procédure générale et détaillée de préparation de la base de données (voir figure 2.2). Elle consiste en plusieurs étapes que nous classifions en deux types majeurs de traitement : (1) nettoyage de la base de données et (2) transformations des données.

2.3.1 Nettoyage de la base de données

Cette étape inclut trois activités : le pré-nettoyage ou la validation des données, le traitement des valeurs aberrantes et la sélection des facteurs d'effort.

2.3.1.1 Pré-nettoyage

Le pré-nettoyage de la base de données consiste à :

a. Vérifier la pertinence des différents attributs de projets et supprimer ceux qui ne sont pas utiles pour le contexte de l'estimation de l'effort. Ces attributs peuvent être des attributs collectés à la fin de projets et qui ne peuvent pas être évalués et utilisés dans l'estimation de l'effort, comme par exemple "la densité des défauts". Les attributs non pertinents peuvent être aussi des attributs relatifs à la collecte de la base de données et les standards utilisés. S'il s'agit de l'estimation de l'effort, alors les attributs dérivés de l'effort sont aussi à supprimer afin d'éviter la redondance, tels que les coûts, les durées, etc. Cette étape s'appuie sur une connaissance basique du problème de l'estimation.

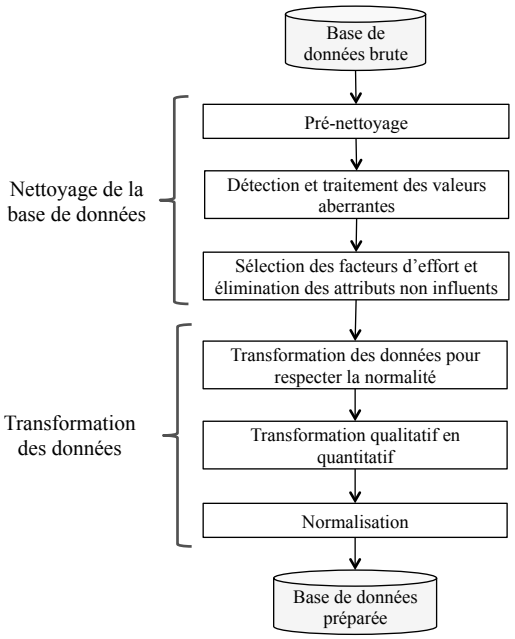


FIGURE 2.2 – Procédure de préparation de la base de données

b. Vérifier la validité des données, c'est à dire leur cohérence, et appliquer des filtres pour supprimer les données non valides, par exemple les projets avec un effort total nul. Les filtres consistent en des règles de logique que l'utilisateur établit sur les attributs de la base de données, par exemple "l'attribut doit être strictement positif".

c. Supprimer les projets dont la qualité de l'information est évaluée comme insuffisante.

d. Identifier et supprimer les attributs qui présentent un fort pourcentage de valeurs manquantes, par exemple un pourcentage supérieur à 40%.

2.3.1.2 Détection et traitement des projets aberrants

Les valeurs aberrantes sont définies comme étant "des observations qui semblent dévier de manière importante des autres observations de la population de laquelle elles proviennent. Ces observations semblent être inconsistantes avec le reste des données" [Everitt2010]. Il est important de les identifier et de les traiter car elles conduisent à de grandes erreurs résiduelles dans les modèles construits. L'identification des observations aberrantes est une tâche difficile et critique surtout quand il s'agit de données multivariées (multidimensionnelles) ce qui est le cas dans notre étude. En effet, contrairement au cas uni-varié où les valeurs aberrantes sont toujours des valeurs extrêmes de l'échantillon, dans le cas multivarié, les valeurs aberrantes sont moins apparentes intuitivement. Elles sont effectivement cachées dans la masse de données et se trouve en périphérie du nuage de points [Afifi+1979 ; Planchon2005]. Les techniques de détection ne sont pas assez développées pour ce cas, nous en citons deux : la distance de Cook et la distance Mahalanobis [Masateru Tsunoda2010].

Pour détecter les projets aberrants de la base de données en prenant en considération les corrélations des données, nous choisissons d'utiliser la distance Mahalanobis D^2 qui peut être également utilisée dans sa version robuste [Hardin+2005]. Elle mesure la distance d'un projet i, $(1 \leq i \leq N)$ du centroïde (moyenne multidimensionnelle) de l'échantillon de projets logiciels (base de données) selon la formule :

$$D_i^2 = (R_i^T - \bar{R}^T)^T \Sigma^{-1} (R_i^T - \bar{R}^T) \tag{2.3.1}$$

où $\bar{R}^T$ représente le vecteur de valeurs moyennes empiriques des attributs et de l'effort de la base de données, $\bar{R}^T = (\bar{A}_1, \bar{A}_2, \cdots, \bar{A}_L, \bar{E})$. Σ^{-1} représente la matrice de variance-covariance de la base de données.

Les projets aberrants qui sont les projets maximisant le distance D^2 sont analysés et traités. Généralement, ils sont supprimés avant d'appliquer un modèle d'estimation. Toutefois, les projets aberrants peuvent contenir des informations importantes et, par conséquent, leurs causes doivent être analysées en détail avant d'effectuer la suppression de ces projets de la base de données.

Il faut noter que l'utilisation de la distance Mahalanobis pour l'identification des projets aberrants se base sur l'hypothèse que les données suivent une loi normale. Il faut donc effectuer une vérification de normalité avant de procéder à l'analyse de projets aberrants. Pour des données non normales, il est possible de les transformer en données normales en utilisant des techniques telles que le Box-Cox (voir 2.3.2.1). Ces transformations consistent à appliquer des coefficients sur les données n'altérant pas leur nature.

2.3.1.3 Sélection des facteurs d'effort

Un facteur d'effort est définit comme une variable prédictive de (très corrélée avec) l'effort [Yu+2003]. L'ensemble des facteurs d'effort est un sous ensemble de $\mathscr{A}$, il peut être représenté par $\mathscr{X} = \{X_i\}\ (1 \leq j \leq e)$, avec e représente le nombre des facteurs d'effort.

La sélection de facteurs d'effort, appelée "sélection de caractéristiques" en statistique, vise à sélectionner, parmi un ensemble de variables, celles qui sont pertinentes, décrivent le mieux la sortie (effort dans notre cas) et optimise la performance du modèle d'estimation.

L'utilisation de tous les attributs de la base de données pré-nettoyée comme variables indépendantes dans la construction du modèle d'estimation peut induire un sur-apprentissage ou sur-ajustement (en anglais "overfitting") [2] de modèle. Ceci est dû au fait que le nombre d'attributs peut être largement supérieur au nombre de projets. Il est alors important de sélectionner les facteurs d'effort et de supprimer les autres attributs ainsi que les attributs redondants (fortement corrélés avec un ou plusieurs attributs). Quand deux attributs sont fortement corrélés entre eux, il suffit d'utiliser un seul. L'utilisation des deux attributs ne fournit pas d'information supplémentaires mais introduit des bruits dans le modèle d'estimation. De surcroit, en ne gardant que les attributs qui ne sont pas corrélés entre eux, le nombre de facteurs d'effort est réduit ainsi que la complexité du modèle, ce qui peut conduire à améliorer la performance du modèle d'estimation.

Ainsi, pour la sélection de facteurs d'effort, nous proposons deux étapes :

— Présélection : nous utilisons des méthodes de filtrage qui permettent de classer les variables en utilisant des critères et de retenir les variables les mieux classées. Nous choisissons comme critère "le coefficient de corrélation" entre l'effort et chacun des attributs de projets. Pour cela, nous utilisons les tests de corrélation de Spearman [Hauke+2011] pour les attributs numériques et One-way ANOVA (F-test) [Surendiran+2010] pour les attributs catégoriques. Ces méthodes de filtrage sont utilisées pour leur simplicité, rapidité et bonne performance [Chandrashekar+2014]. De surcroit, elles sont indépendantes du modèle à construire. Cependant, ces méthodes sont uni-variées, c'est à dire, elles n'examinent pas les relations et corrélations entre les attributs [Guyon+2003 ; Saeys+2007]. C'est pourquoi, une sélection affinée prenant en compte les relations entre les attributs est nécessaire. Nous retenons les attributs numériques ayant un coefficient de Spearman supérieur à 0.4 et les attributs qualitatifs ayant une valeur de P-value inférieur à 0.1 (niveau de signification).

— Sélection affinée : nous utilisons une méthode basée sur les forêts d'arbres décisionnels. C'est l'une des méthodes les plus fiables et plus communément utilisées pour évaluer l'importance/utilité des attributs.

2. Le sur-apprentissage ou sur-ajustement caractérise une situation dans laquelle le modèle d'estimation reproduit avec une grande précision les données d'apprentissage (mémorise la base de données d'apprentissage) tout en étant incapable de faire des estimations performantes pour de nouveaux cas. Ceci survient lorsque la complexité d'un modèle est trop grande par rapport à la relation f qu'il vise à appréhender [Lemberger+2015].

Cette méthode implique la construction d'une forêt d'arbres de décision, c'est à dire, plusieurs modèles (arbres de décision). L'importance d'un attribut est calculée en fonction du nombre et les positions d'apparition de cet attribut dans les arbres de décision. Un attribut qui apparait dans la plupart des arbres et souvent dans des hauts niveaux est considéré très important et ayant une grande influence sur la performance du modèle d'estimation à construire.

Dans cette méthode d'évaluation d'importance d'attributs, chaque attribut est évalué dans un contexte où d'autres attributs sont présents, ainsi cette méthode est similaire aux méthodes enveloppantes qui permettent de détecter les interactions possibles entre les attributs [Cichosz2015].

A la suite de cette étape, les attributs de projet qui ne sont pas considérés influents par l'utilisateur sont supprimés de la base de données.

2.3.2 Transformations des données

Les transformations de données permettent d'assurer le respect des hypothèses requises par certains modèles d'estimation, d'homogénéiser la base de données, et de la rendre adaptée pour la construction/évaluation de modèles d'estimation.

2.3.2.1 Transformation pour respecter la normalité

Certains modèles d'estimation, tels que le modèle RLM sont basés sur des hypothèses fortes [Sakia1992] :

— la population d'observation est normalement distribuée,

— la population d'observation a une variance commune et une structure additive d'erreur

Par conséquent, certains tests statistiques doivent être effectués sur les données pour assurer la validité de ces hypothèses. Selon de nombreuses études [Razali+2011], le test de normalité le plus puissant est le test de Shapiro-Wilk. Dans le cas de violation de ces hypothèses, certaines techniques de transformation peuvent être appliquées, notamment la technique paramétrique de transformation puissance de Box et Cox sélectionnée pour ce travail. La forme originale de la transformation de Box-Cox tel que présentée dans [Box+1964] est définie dans l'équation (2.3.2), où y_i est une valeur d'une variable non normale (attribut ou effort) y du projet i ($1 \leq i \leq N$), $B(y_i, \lambda_b)$ est la transformée de y_i par un paramètre de puissance λ_b. La valeur de λ_b est obtenue en testant différentes valeurs dans $[-5,5]$ et en choisissant celle qui optimise la normalité de y.

$$B(y_i, \lambda_b) = \begin{cases} \frac{y_i^{\lambda_b} - 1}{\lambda_b} & ; \lambda_b \neq 0 \\ log(y_i^{\lambda_b}); \lambda_b = 0 \end{cases} \tag{2.3.2}$$

2.3.2.2 Transformation des variables qualitatives en quantitatives

Certains facteurs d'effort peuvent être qualitatifs, c.à.d. prenant des modalités qualitatives disjointes. Ce type de variables n'est pas pris en considération et représentable par tous les modèles d'estimation, c'est le cas des modèles RLM et RNA. Il est alors nécessaire de transformer les facteurs d'effort qualitatifs en variables quantitatives. A cette fin, nous associons, à chaque facteur d'effort qualitatif, des variables quantitatives binaires représentant chacune une modalité parmi les modalités possibles de ce facteur. Ces nouvelles variables quantitatives sont appelées "variables dummy" ou "variables muettes" [Garavaglia+1998].

Considérons à titre d'exemple la variable qualitative z="type de développement" pouvant prendre deux modalités : "nouveau développement" et "amélioration" . Cette variable peut alors être

représentée par un vecteur de deux composantes dichotomiques : type 1 et type 2, avec :

$$type\ 1 = \begin{cases} 1 & si \quad z = "nouveau\ développement" \\ 0 & sinon \end{cases}$$

$$type\ 2 = \begin{cases} 1 & si \quad z = "amélioration" \\ 0 & sinon \end{cases}$$

2.3.2.3 Normalisation

Les facteurs d'effort de la base de données présentent parfois des plages de valeurs d'échelles différentes. Certains modèles d'estimation sont très sensibles à ces différences d'échelles, notamment les RN. Ils considèrent que l'amplitude de la plage de valeurs d'un attribut est proportionnelle à son importance ce qui n'est pas toujours le cas. Il est donc important d'effectuer un traitement sur les données afin d'assurer une influence identique des facteurs d'effort sur la construction de modèle d'estimation, indépendamment de leurs échelles ou leurs unités de mesures. Pour se faire, les valeurs que prennent les facteurs d'effort dans la base de données sont normalisées pour avoir toutes la même échelle, selon la formule suivante :

$$x_{ij} \leftarrow \frac{x_{ij} - m}{M - m} \quad (1 \leq i \leq N), \quad (1 \leq j \leq e) \tag{2.3.3}$$

où $m = \min_{1 \leq i \leq n}(x_{ij})$ et $M = \max_{1 \leq i \leq n}(x_{ij})$.

2.4 Évaluation de la performance prédictive des modèles d'estimation sur une base de données

Notons respectivement $\mathscr{M}_p$ et $\mathscr{M}_h$ l'ensemble des modèles d'estimation paramétriques et non paramétriques candidats.

Un modèle d'estimation d'effort peut être exprimé de différentes façons selon qu'il est paramétrique (équation 2.4.1a) ou non (équation 2.4.1b).

$$M = \begin{cases} f(x, \beta) & M \in \mathscr{M}_p \quad (2.4.1a) \\ f(x, h, \beta) & M \in \mathscr{M}_h \quad (2.4.1b) \end{cases}$$

Où f est une fonction mathématique qui modélise la relation entre l'effort et les facteurs d'effort. Elle peut être de différents types, RLM ou RNA par exemple. x représente le vecteur des facteurs d'effort. β et h représentent, respectivement, le vecteur des coefficients et le vecteur ou combinaison de valeurs d'hyper-paramètres du modèle d'estimation d'effort.

Cette section vise à proposer une procédure d'évaluation fiable et non biaisée de la performance prédictive des modèles d'estimation candidats.

Deux cas sont à étudier :

— Les modèles d'estimation paramétriques.

— Les modèles d'estimation non paramétriques.

2.4.1 Modèles d'estimation paramétriques d'effort

Pour l'évaluation des modèles d'estimation paramétriques d'effort, nous utilisons la Validation Croisée "K-folds" (VCK).

La procédure de VCK a été décrite dans le chapitre précédent dans la figure 1.11. Elle consiste tout d'abord à partitionner aléatoirement la base de données préparée S en k blocs disjoints. L'ensemble des opérations réalisées sur les blocs $S_i (1 \leq i \leq k)$ est exprimé dans l'algorithme 1 sous la forme d'une fonction appelée VCK. Pour chaque itération i, M est entrainé sur $k-1$ blocs

$\bar{S}_i = S \backslash S_i$. Le modèle calibré sur $\bar{S}_i$ est noté M^i et appelé modèle de substitution. Ce modèle est ensuite testé sur le bloc de test S_i, c'est à dire utilisé pour estimer les efforts des projets de test. La performance P de M^i est calculée sur l'ensemble des projets de S_i en comparant les efforts estimés $\widetilde{E}(S_i)$ aux efforts réels $E(S_i)$. A la fin des k itérations, la performance estimée de M sur la base S, $\widetilde{P}(M,S)$, est obtenue en moyennant les performances des modèles de substitution. La performance finale évaluée par VCK est considérée comme la performance réelle du modèle d'estimation d'effort M calibré sur la base de données préparée entière.

Algorithme 1 : Fonction de validation VCK d'un modèle d'estimation

Function *VCK (M, $S_i (1 \leq i \leq k)$)*
- **Entrée** : Modèle d'estimation M à évaluer
  Base de données S divisé aléatoirement en k blocs $S_i, 1 \leq i \leq k$
- **Sortie** : Performance de M évaluée (estimée) sur S
- **pour** $i = 1, \cdots, k$ **faire**
  - Apprentissage de M sur $\bar{S}_i = S \backslash S_i$: $M^i = M(\bar{S}_i)$;
  - Test du M^i sur S_i et calcul de performance $P(M^i, S_i)$;
- Calcul de la performance globale : $\widetilde{P}(M,S) = \frac{1}{k_{ext}} \sum_i P(M^i, S_i)$;

Dans l'évaluation d'un ensemble de modèles d'estimation paramétriques, il est important d'assurer la conformité, la comparabilité et la répétabilité des résultats de l'évaluation. La conformité et la comparabilité peuvent être assurées en utilisant la même partition (découpage) de la base de données dans la VCK de tous les modèles d'estimation paramétriques. La répétabilité des résultats de l'évaluation, quant à elle, peut être assurée en fixant la génération aléatoire des k blocs.

L'algorithme 2 synthétise la procédure que nous proposons pour l'évaluation d'un ensemble de modèles d'estimation paramétriques.

Algorithme 2 : Algorithme d'évaluation des modèles paramétriques par VCK

Entrée : Base de données préparée S
Ensemble de modèles d'estimation paramétriques candidats $\mathscr{M}_p$
Sortie : Performance de chaque $M \in \mathscr{M}_p$ évaluée (estimée) sur S
Division aléatoire de S en k blocs disjoints ;
pour $M \in \mathscr{M}_p$ **faire**
- Application de la fonction VCK sur M : $\widetilde{P}(M,S) = VCK(M, S_i (1 \leq i \leq k))$;

2.4.2 Modèles d'estimation non paramétriques d'effort

Dans la plupart des travaux analysés dans la littérature, la technique VCK est utilisée aussi bien dans l'évaluation des modèles d'estimation non paramétriques que dans l'évaluation des modèles d'estimation paramétriques. Un modèle d'estimation non paramétrique $M \in \mathscr{M}_h$ se caractérise par des hyper-paramètres qui font que la forme exacte de ce modèle est inconnue avant la sélection des valeurs optimales d'hyper-paramètres. Généralement, plusieurs combinaisons de valeurs d'hyper-paramètres sont possibles. Nous désignons l'ensemble de ces combinaisons par $\mathscr{H}$. A titre d'exemple, dans le cas des modèles d'estimation FAD, les hyper-paramètres peuvent être le nombre de facteurs d'effort sélectionnés aléatoirement et utilisés dans chaque nœud des arbres construits (N_f) et le nombre d'arbres décisionnels qui forme la forêt (N_a). Si nous supposons que $N_f \in \{3,5\}$ et $N_a \in \{200,500\}$ alors $\mathscr{H}$ est l'ensemble des paires (N_f, N_a) : $\mathscr{H} = \{(3,200),(3,500),(5,200),(5,500)\}$.

Pour chaque combinaison possible de valeurs des hyper-paramètres que prend h de $\mathscr{H}$, un modèle paramétrique M_h est instancié à partir de M. Nous appelons M_h un "sous-modèle" de M.

Ainsi, un modèle d'estimation non paramétrique est équivalent à un ensemble de sous-modèles paramétriques possibles (équation 2.4.2).

$$M \in \mathscr{M}_h \Rightarrow M = \{M_h = f_h(x,\beta); \quad h \in \mathscr{H}\} \tag{2.4.2}$$

Ainsi, l'évaluation d'un modèle d'estimation M non paramétrique s'appuie sur l'évaluation des sous modèles M_h associés.

2.4.2.1 Validation croisée "k-fold" pour l'optimisation des hyper-paramètres et l'évaluation de modèles d'estimation non paramétriques

La technique VCK dans le cas d'un modèle d'estimation non paramétrique permet d'une part de déterminer les hyper-paramètres optimaux et d'autre part d'évaluer ce modèle d'estimation.

La procédure de la VCK dans ce cas-ci est représentée dans l'algorithme 3. Elle consiste tout d'abord à évaluer les sous-modèles possibles M_h de M en utilisant la fonction VCK (algorithme 1). Ensuite, le vecteur des valeurs des hyper-paramètres optimaux de M, noté h^*, est déterminé. Il correspond à la combinaison de valeurs d'hyper-paramètres qui optimise la performance de M, c.à.d. qui est associé au sous-modèle optimal. Enfin, la performance du modèle d'estimation M entrainé sur toute la base de données est déduite, elle est considérée équivalente à la performance du sous-modèle optimal (performance maximale). Cette hypothèse est optimiste, elle conduit à une sur-estimation de la performance réelle du modèle d'estimation et peut induire ainsi des biais.

Algorithme 3 : Algorithme d'optimisation des hyper-paramètres et d'évaluation d'un modèle non paramétrique par VCK

Entrée : Modèles d'estimation non paramétrique $M \in \mathscr{M}_h$
Base de données S
Sortie : Performance de M évaluée (estimée) sur S
Combinaison optimale d'hyper-paramètres h^*
Division aléatoire de S en k blocs disjoints ;
Définition de l'ensemble $\mathscr{H}$ des combinaisons possibles de valeurs d'hyper-paramètres ;
pour $h \in \mathscr{H}$ **faire**
  Appliquer la fonction VCK (algorithme 1) sur le modèle M_h :
  $\widetilde{P}(M_h, S) = VCK(M_h, S_i(1 \leq i \leq k))$;
Sélection de la combinaison optimale d'hyper-paramètres : $h^* = \underset{h \in \mathscr{H}}{\arg\min}(\widetilde{P}(M_h, S))$;
Déduction de la performance globale de M : $\widetilde{P}(M, S) = \widetilde{P}(M_{h^*}, S)$;

2.4.2.2 Biais de la validation croisée "k-fold" appliquée sur les modèles d'estimation non paramétriques

Nous appelons les biais induits par la validation croisée "k-fold" appliquée sur les modèles d'estimation non paramétriques, "biais de sélection de sous-modèle", également désignés par "biais de sélection d'hyper-paramètres".

Les biais de sélection sont engendrés par le manque d'indépendance entre les données utilisées pour la construction du modèle d'estimation et les données de test. En effet, pour une évaluation fiable de la performance d'un modèle d'estimation, le modèle construit doit être testé et évalué sur des données de test qui sont indépendantes de celles d'apprentissage. Dans le cas de la procédure VCK pour les modèles non paramétriques, bien que les données de test ne sont pas utilisées dans l'apprentissage du modèle, elles sont utilisées dans la conception du modèle (c.à.d. la sélection des hyper-paramètre). Ainsi, les biais de sélection sont principalement dûs au manque d'indépendance entre l'optimisation des hyper-paramètres et l'évaluation du modèle d'estimation [Varma+2006].

Une autre source des biais de sélection peut être liée à la sélection d'un sous-modèle optimal qui a été sujet au sur-apprentissage (ou sur-ajustement). Le sur-apprentissage peut survenir si la structure du sous-modèle entrainé est trop complexe pour le problème ou pour la quantité de données disponibles (grand nombre de neurones cachés, par exemple). Ce phénomène conduit au stockage de l'information par le sous-modèle et la perte du pouvoir de prédiction de nouveaux cas. Le sous-modèle sur-ajusté mémorise la base de données au lieu d'apprendre à généraliser ses caractéristiques. Il montre alors une performance optimiste qui ne reflète pas son pouvoir de prédiction réel sur de nouveaux cas.

Il est donc important, pour avoir une estimation fiable de la performance réelle d'un modèle d'estimation non paramétrique, d'avoir une indépendance entre la sélection des hyper-paramètres et l'évaluation de la performance du modèle d'estimation.

2.4.2.3 Validation croisée imbriquée des modèles non paramétriques

La Validation Croisée Imbriquée VCI, aussi appelé "validation croisée double", est une évolution de la VCK qui permet une évaluation plus fiable et moins biaisée. C'est une technique statistique de ré-échantillonnage qui a été récemment développée (en 2005) pour réduire les biais d'évaluation de modèles de prédiction. Elle a été utilisée principalement dans le domaine de la bio-informatique [Baumann+2014 ; Statnikov+2005 ; Varma+2006].

La VCI permet de séparer et rendre indépendantes la sélection d'hyper-paramètres et l'évaluation du modèle d'estimation. Son principe repose sur deux boucles de VCK imbriquées, une boucle intérieure permettant l'optimisation des hyper-paramètres du modèle d'estimation et une autre extérieure permettant d'évaluer sa performance. A l'heure actuelle, cette technique n'a pas encore été exploitée dans le domaine de l'estimation de l'effort de développement logiciel. Dans cette étude, nous l'utilisons pour assurer une évaluation fiable de la performance des modèles d'estimation d'effort non paramétrique.

La figure 2.3 décrit la procédure de VCI déployée dans notre étude. Elle se base sur une première division de la base de données en k_{ext} blocs, ensuite des divisions des combinaisons de $(k_{ext}-1)$ blocs en k_{int} blocs inter. La procédure comprend six étapes majeures :

1. La base de données préparée est divisé en k_{ext} blocs externes. A chaque itération $i (1 \leq i \leq k_{ext})$ de la boucle extérieure, $(k_{ext}-1)$ blocs sont utilisés comme données d'apprentissage et le bloc restant est réservé pour le test.

2. Une boucle de VCK (voir algorithme 1) intérieure avec $k = k_{int}$ est effectuée sur les parties d'apprentissage pour chaque combinaison de valeurs d'hyper-paramètres $h \in \mathcal{H}$ afin d'évaluer les sous-modèles d'estimation possibles.

3. Les hyper-paramètres optimaux sont sélectionnés, c'est la combinaison de valeurs h (associée au sous-modèle) qui fournit la meilleure performance sur les données d'apprentissage par la VCK.

4. Le sous-modèle d'estimation optimal (correspondant au modèle d'estimation avec les hyper-paramètres optimaux) est entrainé sur toute la base d'apprentissage.

5. La performance du sous modèle d'estimation optimale est évaluée sur la partie de test.

6. A la fin de la boucle de VCK extérieure, la performance du modèle d'estimation est calculée en faisant la moyenne des performances sur les k_{ext} blocs de test.

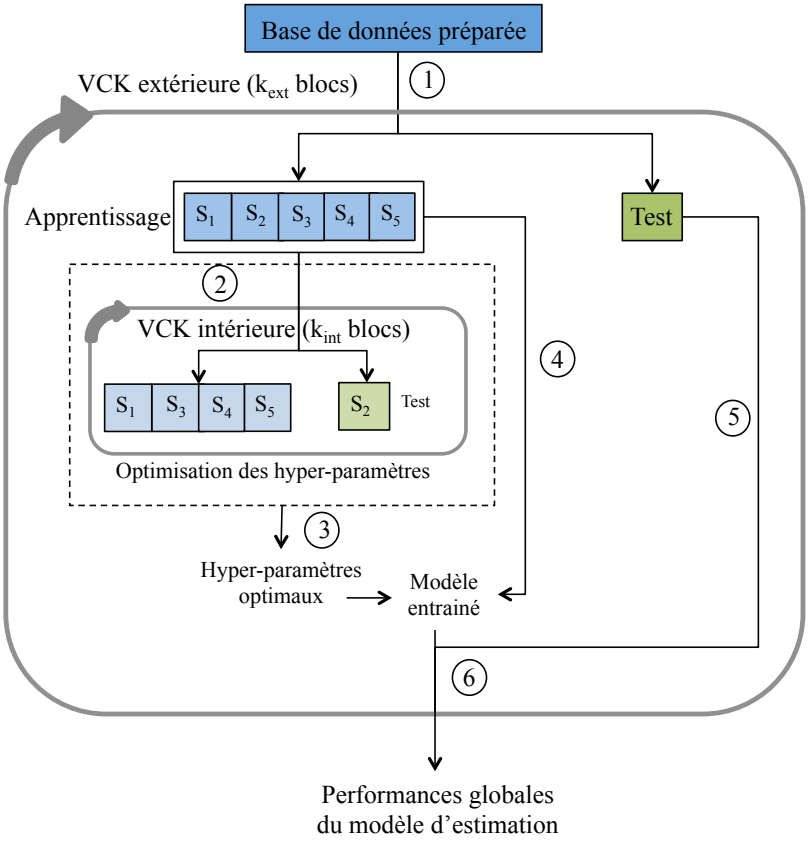


FIGURE 2.3 – Procédure de la validation croisée imbriquée (VCI)

D'une façon similaire à l'évaluation des modèles paramétriques par la VCK, plusieurs indicateurs de performance peuvent être utilisés permettant ainsi une évaluation multi-critère et une flexibilité de sélection du meilleur modèle d'estimation selon les objectifs. Toutefois, dans la boucle intérieure de la VCI, il est plus pratique et approprié d'utiliser un seul indicateur de performance pour faciliter et accélérer la sélection des valeurs optimales d'hyper-paramètres. Cette sélection se fait d'une manière automatisée et interne. Ainsi, nous proposons d'utiliser la mesure d'erreur *MdMRE* comme critère d'optimisation des hyper-paramètres dans la boucle intérieure étant donnée qu'elle est peu sensible aux valeurs extrêmes des erreurs *MRE*. Le *MdMRE* d'un sous-modèle M_h est désigné par e_h.

Le pseudo-code de la technique VCI est également présenté dans l'algorithme 4. La procédure de validation VCI implique un nombre d'opérations (et par conséquent un temps de calcul) relativement élevé par rapport à la VCK. En raison de l'utilisation de deux boucles de validation imbriquées, la procédure de VCI implique l'apprentissage d'un grand nombre de modèles d'estimation. Pour illustrer ceci, supposons que nous voulons appliquer la VCI (avec $k_{ext} = 3$ et $k_{int} = 5$) sur un modèle d'estimation non paramétrique ayant trois combinaisons possibles de valeurs d'hyper-paramètres et une base de données de 500 projets. Si l'apprentissage s'effectue sur 100 itérations d'ajustement. Alors, le nombre d'opérations effectuées dans la VCI est égale à 501 500, alors qu'il est égal à 100 500 dans la VCK. Ainsi, le nombre d'opérations effectuées dans la VCI est cinq fois plus que le nombre d'opérations effectuées dans la VCK dans ce cas. Plus le nombre d'itération de la boucle intérieure augmente plus le nombre d'opérations effectuées dans la VCI devient important.

Algorithme 4 : Algorithme d'évaluation des modèles non paramétriques par VCI

Entrée : Base de données préparée S
Ensemble de modèles d'estimation non paramétriques candidats $\mathscr{M}_h$
Sortie : Performance de chaque $M \in \mathscr{M}_h$ évaluée (estimée) sur S
Division aléatoire de S en k_{ext} blocs disjoints
pour $M \in \mathscr{M}_h$ **faire**
  Définition de l'ensemble $\mathscr{H}$ des combinaisons possibles de valeurs d'hyper-paramètres ;
  pour $i = 1, \cdots, k_{ext}$ **faire**
    pour $h \in \mathscr{H}$ **faire**
      Division aléatoire de $\bar{S}_i = S \backslash S_i$ en k_{int} blocs disjoints $S_j, 1 \leq j \leq k_{int}$;
      Appliquer la fonction VCK (algorithme 1) sur le sous-modèle M_h :
      $\widetilde{e}_h = \widetilde{P}(M_h, \bar{S}_i) = VCK(M_h, S_j(1 \leq j \leq k_{int}))$;
    Sélection de la combinaison optimale d'hyper-paramètres : $h^* = \underset{h \in \mathscr{H}}{\arg\min}(e_h)$;
    Apprentissage de M_{h^*} sur $\bar{S}_i$: $M^i_{h^*} = M_{h^*}(\bar{S}_i)$;
    Test de $M_{h^*,i}$ sur S_i et calcul des performances $P(M_{h^*}, S_i)$;
  Calcul des performances globales : $\widetilde{P}(M, S) = \frac{1}{k_{ext}} \sum_i P(M_{h^*}, S_i)$;

2.4.2.4 Exemple d'application de VCI

Nous présentons un exemple de l'application de la VCI sur un modèle d'estimation non paramétrique de type RNA. Pour ce modèle, deux hyper-paramètres sont à optimiser : le nombre de neurones de la couche cachée (N_c) et le paramètre de régularisation "weight decay" [3] (λ). Dans cet exemple, nous supposons que quatre combinaisons de valeurs de ces hyper-paramètres (N_c, λ) sont possibles : $\mathscr{H} = \{(4, 0.001), (4, 0.0001), (6, 0.001), (6, 0.0001)\}$. Nous avons fixé k_{ext} à 3 et k_{int} à 2.

Les résultats de cet exemple sont présentés dans la figure 2.4. Les numéros qui y figurent, comme dans le schéma 2.3, situent les étapes de la procédure VCI.

La base de données S est divisée en $S1$, $S2$ et $S3$ pour effectuer la boucle extérieure de la VCI. La première table de la figure 2.4 présente les résultats finaux de cette boucle. Chaque ligne de cette table représente une itération de la boucle extérieure, elle nécessite la détermination des hyper-paramètres optimaux avant de pouvoir calculer la performance. Pour la première itération extérieure (1), la deuxième table présente en détails, les résultats de la boucle intérieure (2) effectuée sur $S1$ et $S2$.

Au niveau de la boucle intérieure, les quatre combinaisons possibles des valeurs d'hyper-paramètres sont évaluées, chacune en utilisant la technique VCK. Les valeurs optimales des hyper-paramètres (minimisant *MdMRE*) obtenues sont : $N_c = 4$ et $\lambda = 0.0001$.

Le sous-modèle d'estimation optimale (avec les hyper-paramètres optimaux) est entrainé sur les blocs S1 et S2 (4) et testé sur le bloc S3. Il présente un *MdMRE* de 0.53 sur cette première itération de la boucle extérieure (5). Cet indicateur montre que le sous-modèle a une performance moyenne (une bonne performance, en terme de *MdMRE*, est équivalente à *MdMRE* $\leq 25\%$).

Les résultats des deux lignes de la première table, représentant les deux itérations restant de la boucle extérieure de la VCI, sont obtenus de la même manière que pour la première ligne.

3. Le weight decay est un paramètre utilisé pour éviter le sur-apprentissage des réseaux de neurones. La technique utilisant ce paramètre est une technique de régularisation, elle s'appelle "la méthode de dégradation des pondérations". Cette technique consiste à multiplier les poids du réseaux de neurones par le weight decay afin de limiter la croissance rapide et importante des poids. Ce paramètre est généralement très petit (0.0001 ou moins) [Moody+1995].

La performance globale (6) en terme de *MdMRE* du modèle RNA sur la base de données étudiée, est la moyenne des performances des sous-modèles (0.53, 0.49, 0.62), elle égale à 55%.

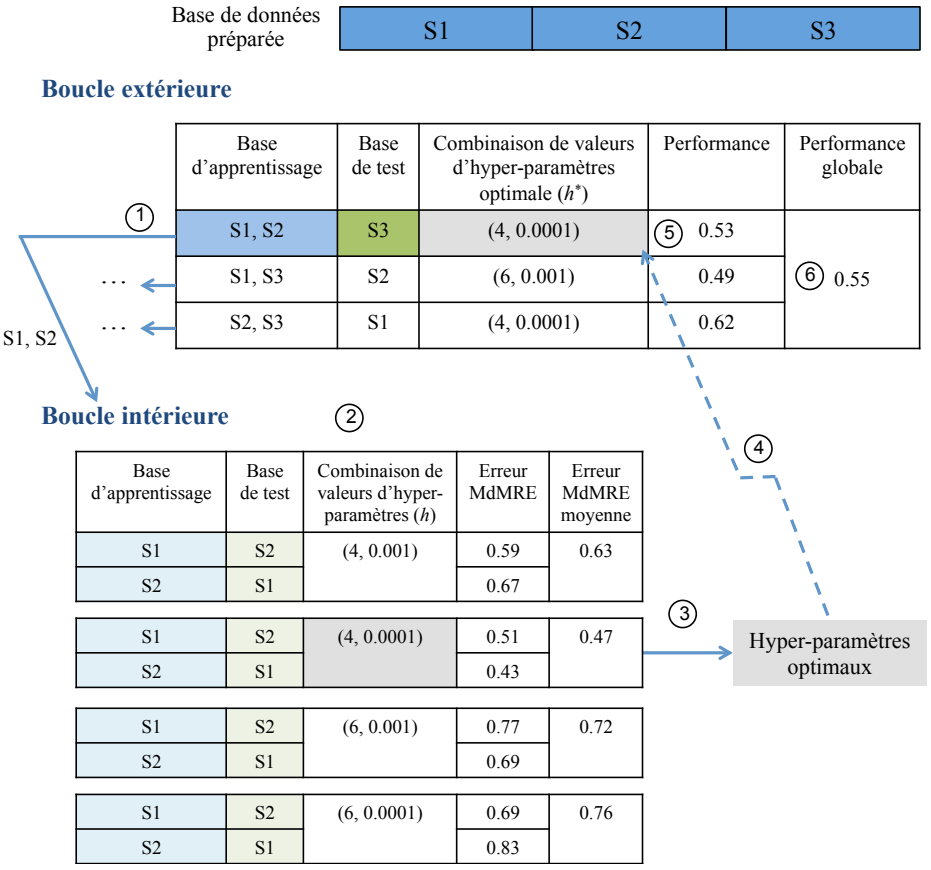


FIGURE 2.4 – Exemple d'application de la VCI sur un modèle d'estimation RNA

2.4.3 Indicateurs de performance prédictive

Nous utilisons différents indicateurs de performance dans l'objectif de représenter et évaluer les aspects importants de la performance prédictive des modèles d'estimation. Ces indicateurs sont : *MMRE*, *MdMRE*, et *Pred*(0.25) (décrits auparavant dans la section 1.4.2).

Nous avons choisi d'utiliser ces indicateurs dans l'évaluation et la comparaison des modèles d'estimation pour différentes raisons. D'une part, du fait de leur utilisation fréquente dans le domaine de l'estimation de l'effort de développement logiciel, les chercheurs et praticiens ont une bonne appréhension de leurs significations. D'autre part, plusieurs travaux de recherche dans le domaine de l'estimation de l'effort de développement l'utilisent, ainsi l'omettre de notre étude restreindra la possibilité de comparaison de nos résultats avec les travaux de recherche existants.

En plus des *MMRE*, *MdMRE* et *Pred*(0.25), nous définissons deux autres indicateurs de performance complémentaires *Und*(l) and *Ovr*(l). Ces indicateurs indiquent, respectivement, la tendance du modèle M à sous-estimer ou sur-estimer l'effort dans un niveau de tolérance l. Ils sont représentés dans l'équation 2.4.3 où u et o sont les nombres de projets ayant une *MRE* inférieure à l et qui, respectivement, sous-estiment et sur-estiment l'effort.

$$Und(l) = \frac{u}{p} \qquad Ovr(l) = \frac{o}{p} \tag{2.4.3}$$

L'ensemble des indicateurs de performance prédictive ne représentent que quelques propriétés de la distribution des *MRE* (par exemple moyenne et médiane). Pour une représentation plus complète du comportement des modèles d'estimation, le diagramme en boîtes des erreurs *MRE* peut être également utilisé (voir figure 1.10). Ce diagramme permet de visualiser d'autres aspects de la distribution des *MRE* tels que les valeurs aberrantes (cas de très grands *MRE*) et la position de 50% des *MRE*.

Après avoir proposé une procédure d'évaluation fiable de la performance prédictive des modèles d'estimation, nous nous intéressons à l'évaluation des incertitudes des modèles d'estimation. L'objectif est de proposer un critère qui représente la mesure des incertitudes et qui peut être utilisé, avec les autres indicateurs de la performance prédictive, dans la sélection de modèle d'estimation adéquat.

2.5 Évaluation des incertitudes des modèles d'estimation sur une base de données

L'objectif de cette section est de proposer une démarche d'évaluation et de quantification des incertitudes d'un modèle d'estimation sur une base de données particulière.

Lorsque nous savons que le modèle d'estimation est entaché d'incertitude (de par sa construction sur la base de données étudiée), il convient d'y faire face en proposant une approche permettant des estimations sous la forme de plages de valeurs, d'IvPs voire de distributions. Ces manières de représentation d'estimation permettent de refléter les incertitudes du modèle d'estimation construit sur la base de données étudiée.

2.5.1 Incertitudes et variabilité

Plus les plages de valeurs des efforts estimés sont étroits, plus le modèle d'estimation construit sur la base de données étudiée est précis et donc moins incertain. Ainsi, la mesure des incertitudes d'un modèle d'estimation calibré sur une base de données est équivalente à la mesure de la variabilité et dispersion des plages de valeurs des efforts estimés. La mesure de variabilité doit être indépendante des biais des estimations d'effort (qui sont les distances entre efforts estimés des efforts réels). Les biais ont été déjà évalués, dans la section précédente, comme aspect de la performance prédictive. Ces concepts d'estimation probabiliste d'effort, de variabilité et de biais sont illustrés dans la figure 2.5.

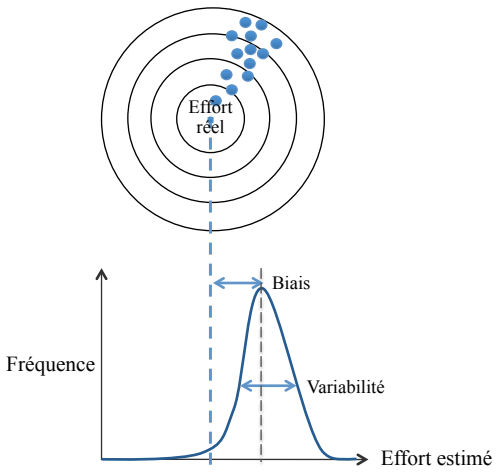


FIGURE 2.5 – Estimation probabiliste de l'effort d'un projet et notion de variabilité et de biais

2.5.2 Bootstrap pour l'évaluation des incertitudes

Afin de pouvoir évaluer les incertitudes d'un modèle d'estimation calibré sur une base de données, notre approche repose sur le couplage d'une technique statistique avec ce modèle. L'idée est de coupler avec le modèle d'estimation à évaluer une technique statistique lui permettant de générer des estimations multiples d'effort (une plage d'estimations) plutôt qu'une estimation unique par projet. La distribution de probabilité des estimations est ensuite analysée afin d'évaluer l'incertitude du modèle d'estimation.

Nous utilisons pour ce propos la technique de bootstrap non paramétrique qui présente l'avantage de ne reposer sur aucune hypothèse concernant la base de données, comme par exemple l'hypothèse que les données suivent une loi de distribution connue et bien déterminée. Cette technique se base sur le ré-échantillonnage pour (1) estimer la performance d'une statistique (un modèle d'estimation dans notre cas) ou (2) estimer la distribution d'une variable (effort dans notre cas).

Étant donné que l'objectif de cette section est d'évaluer les incertitudes de modèles d'estimation calibrés, nous utilisons le principe (1) de ré-échantillonnage de bootstrap.

La procédure que nous proposons pour appliquer sur notre problème est présentée dans la figure 2.6. La technique de bootstrap est appliquée sur la base de données préparée S. B échantillons $S^{*b}, 1 \leq b \leq B$ de même dimension que S sont générés par tirage avec remise à partir de S. La base de données initiale S est utilisée pour le test. Le modèle d'estimation M^{*b} à évaluer est ensuite entrainé sur chaque S^{*b}. Les modèles d'estimation générés (calibrés) $M^{*b}, 1 \leq b \leq B$ sont appelés "répliques" du modèle d'estimation M et sont utilisés pour estimer les efforts des projets de S. Les écarts entre les efforts réels et les efforts estimés sont calculés en utilisant l'indicateur d'erreur *MRE*. Pour chaque itération $b, 1 \leq b \leq B$, le vecteur des erreurs *MRE* est calculé sur S. L'évaluation de l'incertitude repose sur la variabilité évaluée à partir de l'ensemble des vecteurs E_r^{*b}.

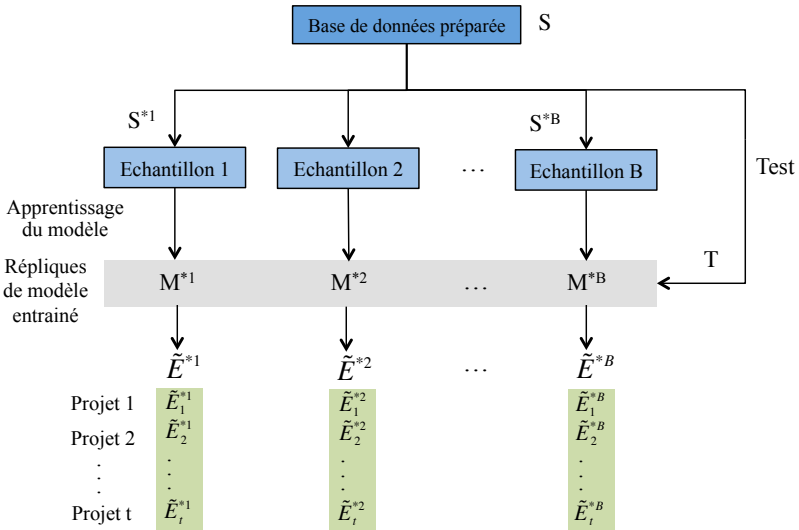


FIGURE 2.6 – Procédure de bootstrap pour l'évaluation de la variabilité de modèles d'estimation d'effort

Pour les modèles d'estimation non paramétriques, il est nécessaire de sélectionner les hyper-paramètres optimaux avant de procéder à l'apprentissage du modèle d'estimation. Dans ce cas, nous proposons d'effectuer l'optimisation des hyper-paramètres sur D en utilisant la VCK comme décrit dans l'algorithme 3 .

L'algorithme 5 représente la procédure d'évaluation de la variabilité des modèles d'estimation candidats pour les deux cas paramétriques et non paramétriques.

Algorithme 5 : Algorithme de bootstrap pour l'évaluation de la variabilité

Entrée : Base de données préparée S
Modèles d'estimation $M \in \mathscr{M}_p$
Sortie :Incertitudes de M évaluées (estimées) sur S
Génération aléatoire avec remise de B échantillons $S^{*b}, 1 \leq b \leq B$ de S ;
pour $M \in \mathscr{M}$ **faire**
  if $M \in \mathscr{M}_h$ **then**
    Sélection des hyper-paramètres optimaux h^* de M par VCK sur S ;
    $M \longleftarrow M_{h^*}$;
  pour $b = 1, \cdots, B$ **faire**
    Apprentissage M sur S^{*b} : $M^{*b} = M(S^{*b})$;
    Application de M^{*b} sur T et calcul des estimations $\widetilde{E}^{*b} = (\widetilde{E}_1^{*b}, \widetilde{E}_2^{*b}, \cdots, \widetilde{E}_n^{*b})$;
  Calcul de la variabilité à partir des $E^{*b} (1 \leq b \leq B)$

2.5.3 Indicateurs d'incertitudes

Les indicateurs de performance généralement utilisés dans la littérature pour l'évaluation des approches probabilistes d'estimation sont : *HitRate*, *RWidth* et *ARPI*. Selon la logique de notre approche qui sépare biais et variabilité, ces indicateurs présentent quelques limitations, à savoir :

— Ils font référence et utilisent les efforts réels. Ainsi, ils ne mesurent pas uniquement la variabilité/incertitudes des estimations mais incluent la mesure de quelques aspects de la performance prédictive (biais).

— Étant donné qu'ils se basent sur les IvPs associés à des degrés de confiance, ils ne représentent la performance que d'une partie de la distribution des efforts estimés et ne prennent pas en compte les valeurs extrêmes.

Pour pallier ces limitations, nous proposons d'évaluer les incertitudes d'un modèle d'estimation calibré en se reposant sur des distributions de l'effort estimé en utilisant des indicateurs de dispersion et de variabilité. Les distributions de probabilité des efforts estimés correspondent aux estimations multiples produites par l'approche basée sur le bootstrap que nous avons proposée.

Il existe plusieurs mesures de dispersion et de variabilité, notamment l'écart moyen, la variance, l'écart type et l'écart type relatif. L'écart type σ est la mesure la plus communément utilisée en statistique, elle se base sur la variance V indiquant la dispersion des observations autour de leur moyenne. L'écart type d'une série d'observations $x_i, 1 \leq i \leq n$ est la racine carrée de sa variance, il se définit comme suit :

$$\sigma = \sqrt{V} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2} \tag{2.5.1}$$

où $\bar{x}$ est la moyenne des observations $x_i, 1 \leq i \leq n$.

Étant donné que l'écart type repose sur le carré des écarts des observations à la moyenne, il est sensible aux valeurs extrêmes ou aberrantes et a tendance à amplifier leurs poids. L'écart type fourni une mesure pertinente de la dispersion dans le cas où la distribution des observations est normale (gaussienne) ou au moins symétrique. Dans le cas contraire, la moyenne des observations peut être déformée et affectée par des valeurs extrêmes et devient non représentative des observations. Dans ce cas de figure, l'écart type n'est pas une mesure fiable et appropriée de la dispersion des observations [Cahan+2011].

Pour pallier ces limitations, nous proposons d'utiliser la Différence Moyenne Relative de Gini (*RMD*). C'est une alternative de mesure de dispersion qui est similaire à la variance relative

mais qui ne fait pas d'hypothèses sur la distribution des observations. Cette mesure est plus pertinente et informative que l'écart type dans le cas d'observations ne suivant pas une loi normale de distribution [Yitzhaki2003]. La mesure *RMD* se base sur les différences entre chaque paire d'observations plutôt que les différences entre chaque observation et la moyenne des observations.

$$RMD = \frac{1}{n}\frac{\sum_{i=1}^{n}|x_i - x_j|}{\bar{x}} \quad i < j \quad (1 \leq i, j \leq n) \tag{2.5.2}$$

Pour un modèle d'estimation, la mesure *RMD* de la dispersion de l'effort estimé, pour un projet k de la base de test, repose sur les estimations d'effort produites en utilisant le bootstrap. Elle se définit ainsi :

$$RMD_k = \frac{1}{B}\sum_{i,j=1}^{B}\frac{|\tilde{E}_k^{*i} - \tilde{E}_k^{*j}|}{\bar{\tilde{E}}_k} \quad i < j \quad (1 \leq i, j \leq B) \tag{2.5.3}$$

où $\tilde{E}_k^{*i}$ et $\bar{\tilde{E}}_k$ sont respectivement, l'effort estimé et la moyenne des efforts estimés du projet k en utilisant la réplique du modèle d'estimation entrainé sur l'échantillon i.

Pour l'évaluation de la dispersion globale des estimations d'un modèle d'estimation, nous définissons la mesure de dispersion globale *MRMD*. Il représente la moyenne des *RMD* sur l'ensemble des projets de test.

$$MRMD = \frac{1}{t}\sum_{k=1}^{t} RMD_k \tag{2.5.4}$$

Cette mesure de dispersion peut être utilisée comme indicateur indirect des incertitudes de modèles d'estimation calibrés sur des bases de données spécifiques. Ainsi, plus le *MRMD* d'un modèle d'estimation d'effort calibré est grand, plus ce modèle d'estimation présente d'importantes incertitudes. La procédure d'évaluation des incertitudes de modèles d'estimation est présentée dans la figure 2.7.

	$\tilde{E}^{*1}$	$\tilde{E}^{*2}$		$\tilde{E}^{*B}$	Dispersion	Dispersion globale
Projet 1	$\tilde{E}_1^{*1}$	$\tilde{E}_1^{*2}$	…	$\tilde{E}_1^{*2}$	RMD_1	
Projet 2	$\tilde{E}_2^{*1}$	$\tilde{E}_2^{*1}$	…	$\tilde{E}_2^{*1}$	RMD_2	$\frac{1}{t}\sum_{k=1}^{t} RMD_k$
⋮	⋮	⋮		⋮		
Projet t	$\tilde{E}_t^{*1}$	$\tilde{E}_t^{*1}$	…	$\tilde{E}_t^{*1}$	RMD_t	

FIGURE 2.7 – Évaluation de l'indicateur d'incertitudes (*MRMD*) d'un modèle d'estimation d'effort

Maintenant que nous avons défini l'ensemble des critères/indicateurs de performance et d'incertitudes, nous nous intéressons dans la section suivante à fournir des orientations aux utilisateurs pour les supporter dans la sélection du modèle d'estimation le plus adéquat à leurs besoins et objectifs.

2.6 Aide à la sélection du meilleur modèle d'estimation d'effort

L'approche d'évaluation que nous proposons caractérise les modèles d'estimation candidats en utilisant différents indicateurs de performance. Ces indicateurs permettent d'avoir une vue globale sur les différents aspects de la capacité prédictive et des incertitudes de modèles d'estimation à

comparer. Cependant, la sélection d'un modèle d'estimation peut être une tâche difficile en raison de cette multitude de critères qui ne varient pas forcément dans le même sens.

Dans cette section, des orientations sont proposées pour supporter les utilisateurs de notre approche dans la sélection d'un modèle d'estimation adéquat (performant). Cette sélection repose sur la stratégie et les besoins propres à chaque utilisateur.

Dans le cas ou l'utilisateur souhaite avoir des estimations généralement stables, c'est à dire des estimations de qualité similaire, le modèle d'estimation avec le plus petit *MMRE* devrait être sélectionné. Le *MMRE* étant une moyenne d'erreurs, il est sensible aux erreurs extrêmes et donc il représente la variabilité des erreurs. Cette stratégie correspond à l'aversion pour le risque, étant donnée qu'elle cherche un modèle d'estimation qui se comporte bien même pour les projets atypiques [Kitchenham+1999].

Si plus d'attention est portée aux projets typiques, il est plus important de produire des estimations stables particulièrement pour ces projets. Dans ce cas, le modèle d'estimation qui minimise la *MdMRE* est plus performant que les autres. En effet, le *MdMRE* est moins sensible aux valeurs extrêmes de *MRE*, ainsi il ne représente pas les estimations aberrantes souvent associées aux projets atypiques. Cette stratégie correspond au goût pour le risque.

D'autres contextes d'aversion pour le risque peuvent exiger un modèle d'estimation qui fournit le plus grand nombre d'estimations performantes. Dans ce cas, il est recommandé de choisir le modèle d'estimation ayant le plus grand *Pred*(0,25).

Face à deux modèles d'estimation ayant des performances similaires (*MMRE*, *MdMRE* ou *Pred*(0,25)), l'utilisateur peut souhaiter favoriser le modèle d'estimation qui a moins de risque de sous-estimation ou au contraire qui a moins de risque de sur-estimation. La sélection du modèle d'estimation, dans ce cas, doit s'appuyer sur *Und* ou *Ovr* selon le niveau d'acceptation des dépassement éventuels d'effort. Par exemple, dans le cas des start-up en développement informatique, il est très important de ne pas dépasser les budgets prévus étant donné que les ressources sont souvent limitées, ainsi l'estimateur peut favoriser, parmi plusieurs modèles d'estimation ayant les mêmes *MMRE*, *MdMRE* ou *Pred*(0,25), le modèle minimisant le *Und* .

Bien que le choix de l'indicateur de la performance prédictive selon la stratégie de l'utilisateur soit une bonne option pour guider la sélection de modèle d'estimation, les objectifs de l'utilisateur sont rarement unidimensionnels. Dans ce cas, l'utilisateur peut opter pour une représentation générale et graphique des erreurs à travers le diagramme en boîtes. Cette représentation permet de visualiser et de confronter les distributions des erreurs des modèles d'estimation candidats, et ainsi faciliter la comparaison des ces modèles. Cependant, elle nécessite une connaissance élémentaire de l'outil afin de pouvoir interpréter les résultats et sélectionner le modèle d'estimation adéquat.

L'indicateur unique d'incertitudes *MRMD* est important pour garantir qu'en mode probabiliste le modèle d'estimation sélectionné produit des estimations réalistes et précises.

La table 2.1 synthétise d'une manière simple les indicateurs de performance en relation avec les stratégies d'estimation adoptées par l'utilisateur et la tendance à chercher lors de la sélection du meilleur modèle d'estimation.

TABLE 2.1 – Synthèse des indicateurs de performances

Critère de sélection	Stratégie	Minimiser ou maximiser
MMRE	Aversion pour le risque	A minimiser
MdMRE	Goût pour le risque	A minimiser
Pred(0.25)	Grand nombre d'estimations performantes	A maximiser
Und(0.25)	Risque de dépassement petit	A minimiser
Ovr(0.25)	Risque de dépassement grand	A minimiser
MRMD	Estimations précises en mode probabiliste	A minimiser

Conclusion

La sélection de modèle d'estimation le plus performant est un enjeu majeur dans le domaine de l'estimation de l'effort des projets logiciels. Avec l'évolution des techniques de modélisation, une panoplie de modèles d'estimation a été proposée dans la littérature.

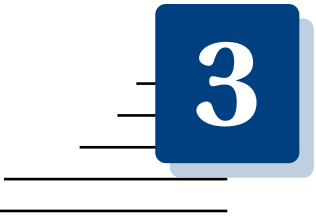
Bien qu'il existe une multitude de travaux de comparaison de ces modèles d'estimation, il est difficile de conclure sur le modèle d'estimation le plus performant pour un environnement de développement particulier. Les résultats de comparaison sont souvent contradictoires en raison des biais induits par les méthodes d'évaluation et validation et la non pertinence de certains indicateurs de performance utilisés. De surcroit, peu de travaux comparent les modèles d'estimation en mode probabiliste et prennent en compte ses incertitudes dans la comparaison et la sélection. Ainsi, la question se pose encore de savoir lequel des modèles d'estimation est le plus adéquat pour construire un système d'estimation performant et réaliste pour une organisation particulière.

Nous avons proposé une procédure fiable d'évaluation de modèles d'estimation qui vise à réduire les biais d'évaluation qui sont importants pour le cas des modèles non paramétriques. La méthode VCK est utilisée pour les modèle paramétriques, tandis que pour les modèles non paramétriques nous avons préconisé l'utilisation de la VCI. C'est une méthode de validation récente permettant de réduire les biais de la VCK pour le cas des modèles non paramétriques. Quoique la VCI présente une limitation liée au temps de calcul élevé qu'elle nécessite, la VCI n'introduit pas une difficulté potentielle dans le contexte de notre étude. En effet, la sélection de modèle d'estimation n'est pas une activité très récurrente. En raison de l'évolution du contexte, de l'environnement de développement et de la base de données de l'organisation, le système d'estimation utilisé peut voir sa performance se dégrader. Par conséquent, l'utilisateur peut décider de ré effectuer la procédure de sélection de modèle d'estimation pour reconstruire ou mettre à jour le système d'estimation d'effort devenu dépassé. Le délai moyen de cette mise à jour peut être estimé entre 6 mois et 1 ans selon l'organisation.

Divers critères de sélection ont été proposés dans notre étude. Ils permettent d'assurer une grande flexibilité de sélection du modèle d'estimation selon les objectifs et les stratégies adoptés par l'utilisateur.

Après la sélection du modèle d'estimation le plus adéquat pour l'organisation, l'étape qui suit est l'utilisation de ce modèle pour construire un système d'estimation d'effort propre à l'organisation, réaliste et performant. Il s'agit d'un système qui est "prêt à l'emploi" et qui permet d'estimer d'une façon réaliste l'effort des nouveaux projets de développement logiciel de l'organisation.

CHAPITRE



Construction et utilisation d'un système d'estimation réaliste dans un environnement incertain

Introduction

L'approche que nous avons proposée dans le chapitre précédent permet de sélectionner un modèle d'estimation pour une organisation particulière. Elle s'appuie sur une comparaison fiable de différents modèles d'estimation en tenant compte des incertitudes. Le modèle d'estimation sélectionné ne peut être utilisé directement pour l'estimation de nouveaux projets. Pour produire des estimations de l'effort de nouveaux projets, il faut calibrer le modèle d'estimation sur la base de données de l'organisation. Nous parlons de la "construction" d'un système d'estimation en s'appuyant sur le modèle d'estimation sélectionné.

Ce système d'estimation doit être **performant**, il doit pouvoir :

a. produire des estimations d'effort représentant la réalité qui se caractérise par la présence des incertitudes dans la phase de construction du système d'estimation (incertitudes du modèle d'estimation sélectionné σ_M et celles de la base de données σ_S). L'estimation ponctuelle n'étant pas représentative de ces incertitudes, le système d'estimation doit permettre des estimations probabilistes (plages de valeurs ou IvPs, voir distributions de probabilité),

b. produire des estimations d'effort les plus proches possible des efforts réels, c'est à dire il doit avoir une bonne performance prédictive.

Les nouveaux projets présentent d'importantes incertitudes σ_P, surtout en phases amont du cycle de vie de projet. Ces incertitudes sont relatives au manque, à l'incomplétude et à l'imprécision des informations sur le projet et aux changements des spécifications et de l'environnement de développement. Pour avoir une estimation réaliste, il est donc important de prendre également en compte ces incertitudes lors de l'estimation de l'effort de nouveaux projets.

La figure 3.1 présente une vue d'ensemble du processus de construction et d'utilisation d'un système d'estimation d'effort.

Ainsi, l'objectif de ce chapitre est de proposer une approche permettant de :

1. construire un système d'estimation d'effort performant propre à l'entreprise à partir du modèle d'estimation sélectionné.
2. utiliser le système d'estimation d'effort construit pour l'estimation de l'effort de nouveaux projets entachés d'incertitudes.

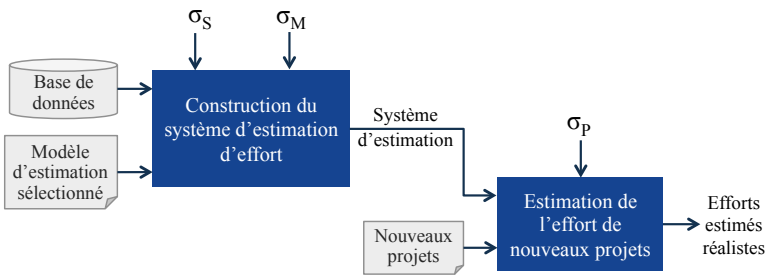


FIGURE 3.1 – Construction et utilisation d'un système d'estimation d'effort dans l'estimation de nouveaux projets

3.1 Construction d'un système d'estimation performant

La construction d'un système d'estimation se base sur le modèle d'estimation sélectionné. Elle consiste à calibrer ce modèle sur la base de données de l'organisation, en d'autres termes, déterminer les paramètres du modèle d'estimation (coefficients et aussi hyper-paramètres dans le cas des modèles d'estimation non paramétriques). Dans l'approche classique, les paramètres sont déterminés d'une façon déterministe, c'est à dire que chaque paramètre consiste en une valeur unique. Ainsi, le système d'estimation construit consiste en un modèle d'estimation unique calibré sur la base de données de l'entreprise.

La phase de construction d'un système d'estimation se caractérise par la présence des incertitudes σ_M et σ_S. Ces incertitudes affectent le système d'estimation et plus précisément ses paramètres. Ces derniers sont entachés d'incertitudes. Une façon de représenter ces incertitudes est d'exprimer chacun des paramètres d'une façon probabiliste sous la forme d'un ensemble de valeurs possibles. Ceci nous conduit à construire un **système d'estimation multi-modèles**, c'est à dire composé de plusieurs modèles d'estimation calibrés sur la base de données de l'organisation. La figure 3.2 illustre par un exemple ces concepts.

Nous proposons d'utiliser pour la construction du système d'estimation multi-modèles la technique de ré-échantillonnage bootstrap. L'idée principale du bootstrap est que l'ensemble des échantillons tirés de la base de données disponible fournit une meilleure estimation et représentation de la distribution de la population d'origine. Ainsi, le système d'estimation doit se baser sur les échantillons de bootstrap afin qu'il soit plus représentatif de la réalité et de ses incertitudes.

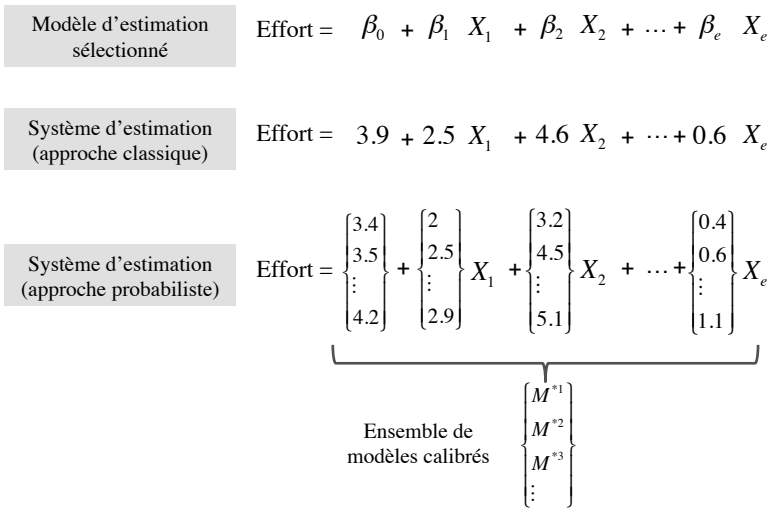


FIGURE 3.2 – Approche classique et approche probabiliste de construction de système d'estimation d'effort

Système d'estimation multi-modèles basé sur le bootstrapp

Étant donné que le modèle d'estimation sélectionné peut être paramétrique ou non paramétrique, la procédure d'application de bootstrap n'est pas uniforme, comme noté dans le chapitre précédent. En effet, dans le cas de modèles d'estimation non paramétriques, une étape d'optimisation et de sélection d'hyper-paramètres est nécessaire. Nous utilisons pour ce propos la VCK (voir algorithme 3 [1]).

Les algorithmes 6 et 7 présentent le processus de construction d'un système d'estimation d'effort, que nous notons $\mathscr{S}$, à partir d'un modèle d'estimation paramétrique et non paramétrique, respectivement.

Algorithme 6 : Algorithme de bootstrap pour la construction d'un système d'estimation d'effort à partir d'un modèle d'estimation paramétrique

Entrée : Modèles d'estimation sélectionné M
Sortie :Système d'estimation d'effort
Génération aléatoire avec remise de B échantillons $S_b^*, 1 \leq b \leq B$ de S (même dimension) ;
pour $b = 1, \cdots, B$ **faire**
  Apprentissage M sur S_b^* : $M^{*b} = M(S_b^*)$;
$\mathscr{S} = \{M^{*b}; b \in 1, \cdots, B\}$

Plus le nombre de bootstrap B est grand, mieux le système d'estimation représente les incertitudes. Il peut être fixé entre 20 et 200 [Efron+1994]. Néanmoins, un très grand nombre de B implique la construction d'un système composé d'un très grand nombre de modèles d'estimation calibrés. Par conséquent, la construction ainsi que l'utilisation du système construit dans l'estimation de nouveau projet nécessiteront un temps de calcul relativement élevé.

En ce qui concerne la performance prédictive du système d'estimation construit, il convient de noter que le système d'estimation est dérivé de la base de donnée entière. Ainsi, son apprentissage

1. Cet algorithme a été initialement conçu pour l'optimisation des hyper-paramètres ainsi que pour l'évaluation des modèles non paramétriques. Dans cette phase du processus d'estimation, nous nous arrêtons à l'étape de sélection de la combinaison optimale d'hyper-paramètres étant donné que l'évaluation du modèle a été déjà effectuée.

Algorithme 7 : Algorithme de bootstrap pour la construction d'un système d'estimation d'effort à partir d'un modèle d'estimation non paramétrique

Entrée : Modèles d'estimation sélectionné M
Sortie :Système d'estimation d'effort
Génération aléatoire avec remise de B échantillons $S_b^*, 1 \leq b \leq B$ de S (même dimension) ;
Sélection des hyper-paramètres optimaux h^* de M par VCK sur S ;
pour $b = 1, \cdots, B$ **faire**
  Apprentissage M_{h^*} sur S_b^* : $M_{h^*}^{*b} = M_{h^*}(S_b^*)$;
$\mathscr{S} = \{M_{h^*}^{*b}; b \in 1, \cdots, B\}$

s'appuie sur plus de données (projets réalisés) par rapport au modèle d'estimation sélectionné. Par conséquent, il s'ajuste mieux à la base de données et montre de bonnes performances.

3.2 Utilisation du système d'estimation dans un environnement incertain

Le processus de spécification d'un nouveau projet de développement logiciel est naturellement sujet aux incertitudes relatives au manque de la connaissance sur ce projet (σ_P). Cela par exemple peut être illustré par la difficulté à donner une mesure exacte de la taille du projet en points de fonction. Ces incertitudes peuvent être représentées dans les facteurs d'effort sous la forme de Fonction de Densité de Probabilité (FDP). L'effort estimé d'un nouveau projet doit représenter aussi bien ces incertitudes (σ_P) que les incertitudes σ_M et σ_S pré-intégrées dans le système d'estimation.

Pour quantifier l'impact de la connaissance imprécise et incomplète du nouveau projet sur l'effort estimé par le système d'estimation, nous proposons d'utiliser la méthode Monte Carlo. Cette méthode implique l'utilisation des FDPs pour exprimer les facteurs d'effort incertains. Elle permet de propager les incertitudes caractérisées des facteurs d'effort dans le système d'estimation pour obtenir une estimation de l'effort faisant apparaitre ces incertitudes (voir figure 3.3).

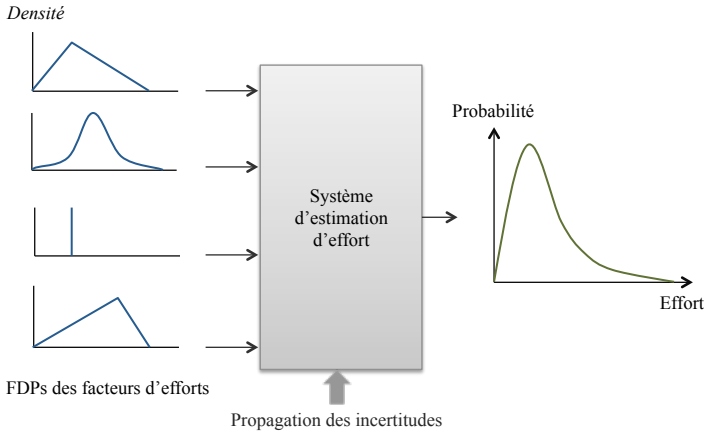


FIGURE 3.3 – Propagation des incertitudes des facteurs d'effort

La méthode Monte Carlo se base sur des nombres aléatoires pour prélever des échantillons des FDPs des facteurs d'effort et exécuter des simulations permettant de propager ces FDPs vers

l'effort estimé [Mooney1997]. Elle a différents avantages, entre autres, elle permet de traiter aussi bien les grandes que les petites incertitudes des valeurs des facteurs d'effort [JCGM2008].

L'approche de propagation des incertitudes relatives aux données d'un nouveau projet consiste en trois étapes majeures :

— Détermination de la FDP de chaque facteur d'effort du nouveau projet à estimer et surtout ceux qui sont incertains,

— Exécution de la simulation Monte Carlo,

— Analyse des résultats de la simulation et expression de l'effort estimé.

3.2.1 Détermination des FDP des facteurs d'effort

Afin de refléter les incertitudes présentes dans l'évaluation des facteurs d'effort d'un nouveau projet à estimer, ces facteurs peuvent alors être exprimés dans le mode probabiliste notamment sous la forme de FDPs. Le choix des FDPs les plus appropriées se fait par l'utilisateur en se basant sur les spécifications du projet ainsi que sur son expérience. Ce choix peut dépendre aussi de la facilité d'implémentation des FDPs et les temps de calcul. Parmi les nombreux types de FDP existants, nous en choisissons cinq qui semblent être les plus importantes et simples à implémenter et à appréhender par l'utilisateur. Il s'agit de : (1) la distribution normale, (2) la distribution triangulaire, (3) la distribution uniforme, (4) la distribution discrète et (5) la distribution booléenne (voir figure 3.4).

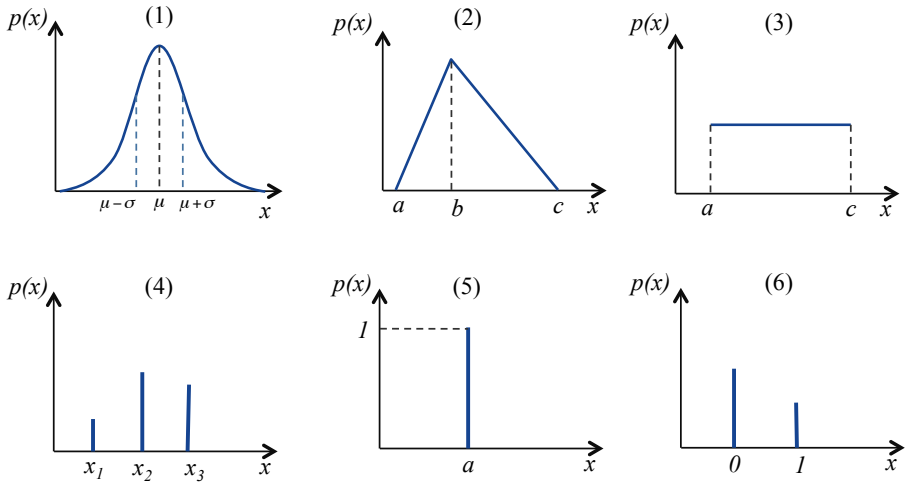


FIGURE 3.4 – Les distributions de probabilité les plus courantes

3.2.1.1 Distribution normale (1)

La distribution normale est une des distributions les plus fréquemment utilisées. Elle est définie par une moyenne notée μ et un écart type noté σ. Une variable x qui suit la loi normale est dite normale ou gaussienne. La FDP associée à cette loi, notée $p(x)$, peut s'écrire comme suit :

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\frac{x-\sigma}{\sigma}^2} \tag{3.2.1}$$

La loi normale décrit souvent des phénomènes naturelles de l'univers physique telles que la diffusion des gaz, et les erreurs de mesures métrologiques. Elle suppose qu'une variable normale provient souvent de la combinaison de plusieurs facteurs aléatoires indépendants, ou à partir d'un processus naturel.

La distribution normale est une des distributions les plus adaptées pour la modélisation des erreurs, par exemple dans l'analyse de la régression des moindres carrées. Dans le cas de l'estimation de l'effort, ce type de distribution peut représenter la taille fonctionnelle du projet en PF.

Afin de modéliser un facteur d'effort suivant la loi normale, l'utilisateur doit spécifier sa moyenne et son écart type. Ce dernier peut être difficile à caractériser selon la maitrise de l'utilisateur de cette modélisation statistique.

3.2.1.2 Distribution triangulaire (2)

C'est la distribution de probabilité la plus pratiquement utilisée pour modéliser l'opinion de l'expert. Elle est définie par une valeur minimale (a), une valeur la plus probable (b) et une valeur maximale (c). La FDP de cette distribution est :

$$p(x) = \begin{cases} \frac{2(x-a)}{(b-a)(c-a)}; & a \leq x \leq b \\ \frac{2(c-x)}{(c-b)(c-a)}; & b \leq x \leq c \\ 0; & x < a \ \ ou \ \ x > b \end{cases} \tag{3.2.2}$$

Cette représentation peut être asymétrique et offre par conséquent la possibilité de représenter une variable déformée et désaxée vers la gauche ou la droite.

La distribution triangulaire est souvent utilisée comme une description subjective de la variable à évaluer. Elle est utilisée comme alternative de la distribution normale étant donnée qu'il est plus simple et facile d'estimer a, b et c que μ et σ. Dans le domaine de l'estimation de l'effort, les experts peuvent favoriser son utilisation facile et simple pour modéliser des facteurs d'effort tels que la taille fonctionnelle du projet.

3.2.1.3 Distribution uniforme (3)

La distribution uniforme est considérée comme une loi d'équiprobabilité, elle est définie par une valeur inférieure (a) et une valeur supérieure (b). Dans cette distribution, la densité de probabilité d'une variable x est constante sur l'intervalle entre a et b, cependant, elle est nulle en dehors de cette intervalle.

$$p(x) = \begin{cases} \frac{1}{b-a}; & a \leq x \leq b \\ 0; & x < a \ \ ou \ \ x > b \end{cases} \tag{3.2.3}$$

Cette distribution se caractérise par sa facilité d'utilisation et d'implémentation. Elle peut être considérée comme un moyen pour obtenir une solution rapide et peut être utilisée pour mettre en exergue le manque d'information sur la variable à évaluer. En effet, la distribution uniforme suppose que l'utilisateur ne connait que les limites inférieure et supérieure du facteur d'effort incertain du projet à estimer et qu'il n'a aucun doute qu'une quelconque valeur soit plus ou moins probable que d'autres.

3.2.1.4 Distribution discrète (4, 5, 6)

La distribution discrète permet de décrire trois types de variables :

— Variables numériques discrètes (4) : Chaque valeur discrète de la variable à évaluer a une probabilité associée de p_i qui indique la probabilité de sa survenance. La somme des probabilité, p_i, doit être égale à 1. La FDP décrivant cette distribution s'écrit :

$$p(x_i) = p_i; \quad x = x_i \tag{3.2.4}$$

Dans les problème de l'estimation d'effort, cette distribution peut représenter par exemple la taille de l'équipe de développement.

— Variables certaines (5) : Il s'agit d'une variable dont l'utilisateur ou l'expert connait la valeur exacte au stade de l'estimation de l'effort. Il peut être représentée par un point unique de

probabilité égale à 1 (voir figure distribution dans la figure 3.4). La FDP associée peut être décrite ainsi :

$$p(x) = \begin{cases} 1; & x = a \\ 0; & x \neq a \end{cases} \tag{3.2.5}$$

— Variables booléennes (6) : La distribution discrète peut aussi être utilisée pour modéliser une variable booléenne. La FDP de cette distribution prend deux valeurs : Vrai (1) et Faux (0). La probabilité qu'une variable est vrai, étant p, la FDP décrivant cette distribution peut s'écrire comme suit :

$$p(x) = \begin{cases} 1-p; & x = 0 \\ p; & x = 1 \end{cases} \tag{3.2.6}$$

Elle peut être utilisée dans le cas des facteurs d'effort qui sont qualitatifs et qui ont été transformés en des variables muettes booléennes. Considérons à titre d'exemple la variable qualitative T="type du langage de développement" pouvant prendre deux modalités : "3GL" et "4GL". Cette variable peut être représentée par un vecteur de deux composantes dichotomiques : T.3GL et T.4GL, avec :

$$T.3GL = \begin{cases} 1; & T = "3GL" \\ 0; & T \neq "3GL" \end{cases}$$

$$T.4GL = \begin{cases} 1; & T = "4GL" \\ 0; & T \neq "4GL" \end{cases}$$

En phase amont de l'estimation d'un nouveau projet , supposons que l'expert n'est pas certain du type de langage à utiliser pour le développement de ce projet. Il estime qu'il y a une probabilité de 70% qu'il soit développé en "3GL". Ainsi la variable T.3GL peut être représentée par la FDP suivante :

$$p(T.3GL) = \begin{cases} 0.3; & T.3GL = 0 \\ 0.7; & T.3GL = 1 \end{cases} \tag{3.2.7}$$

3.2.2 Simulation Monte Carlo

A cette étape, la question qui se pose est de savoir comment estimer l'effort en propageant simultanément tous les facteurs d'effort incertains, représentés par des FDPs, dans le système d'estimation de l'effort. Cette étape nécessite souvent une forte intensité de calcul en raison des simulations numériques répétées de Monte Carlo. Nous détaillons dans la suite, le principe de la méthode Monte Carlo, son adaptation et utilisation dans le contexte de notre étude.

3.2.2.1 Principe de la simulation Monte Carlo

La simulation Monte Carlo consiste à remplacer chaque variable d'entrée incertaine représentée par une FDP par un ensemble d'échantillons aléatoires pouvant donner une approximation de la FDP et permettre le calcul des valeurs de sortie possibles.

La procédure d'échantillonnage des FDP des variables d'entrée est illustrée dans la figure 3.5. A partir de la FDP de la variable d'entrée à échantillonner, une distribution cumulative P_c est générée. L'échantillonnage s'appuie sur un générateur de nombres aléatoires uniformément distribués compris entre 0 et 1 [Herrador+2005]. Chaque nombre aléatoire généré r est projeté à partir de l'axe des P_c sur la courbe de distribution cumulée et la valeur x_r correspondante est relevée. Cette valeur est considérée comme un échantillon de la FDP étudiée. Cet échantillonnage est répété m fois. L'ensemble des échantillons est supposé fournir une approximation de la FDP représentant les incertitudes de la variable d'entrée étudiée. Il est important de noter que plus m est large plus Monte Carlo donne une bonne approximation de la FDP étudiée [Dienemann1966]. Cependant, un très large m implique un temps de calcul élevé. Il est à souligner également l'importance d'un générateur de nombre aléatoire assez performant afin d'obtenir des échantillons fournissant une

bonne approximation de la FDP échantillonné. Dans le cas des facteurs d'effort certains, leurs valeurs restent constantes lors de l'échantillonnage.

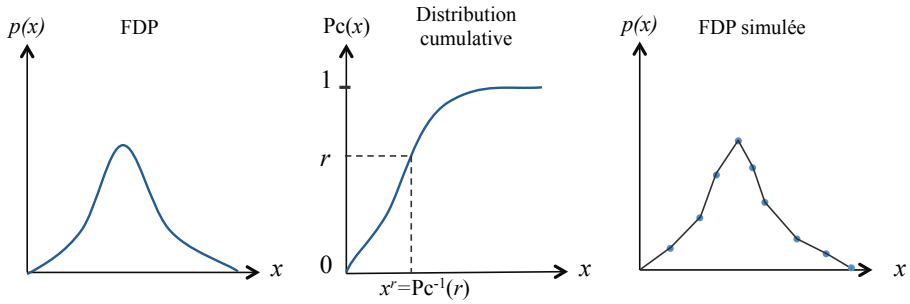


FIGURE 3.5 – Échantillonnage Monte Carlo

Généralement, Monte Carlo est utilisée pour propager les incertitudes dans un modèle unique. La procédure Monte Carlo pour l'estimation d'un nouveau projet, est présentée dans la figure 3.6. Elle consiste en m itérations, à chaque itération r ($r \leq m$) l'échantillonnage de Monte Carlo est effectué sur les facteurs d'effort décrivant ce nouveau projet et exprimées par des FDPs. Nous appelons l'ensemble d'échantillons obtenus "réalisation". Pour chaque réalisation, une valeur unique de l'effort estimé est calculée en utilisant le modèle d'estimation. L'ensemble des m estimations résultant de cette simulation est analysé. Il peut être organisé en un histogramme permettant ainsi d'obtenir la distribution de l'effort estimé et l'IvP de l'effort.

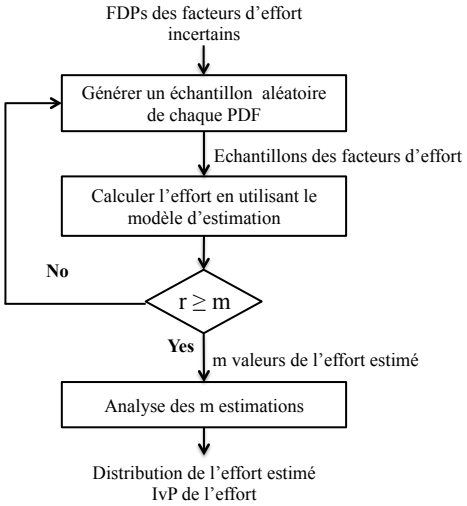


FIGURE 3.6 – Simulation Monte Carlo pour propager les incertitudes d'un nouveau projet dans un modèle d'estimation calibré

3.2.2.2 Monte Carlo appliqué sur un système d'estimation d'effort

Dans le cas de notre étude, il ne s'agit pas d'un modèle unique auquel nous souhaitons appliquer la méthode de Monte Carlo, mais plutôt d'un système composé de plusieurs modèles d'estimation d'effort calibrés. Ainsi, la procédure de Monte Carlo doit être adaptée afin de permettre la propagation des incertitudes des facteurs d'effort du nouveau projet dans le système d'estimation d'effort.

La procédure d'application de Monte Carlo que nous proposons pour la propagation des incertitudes dans un système composé de modèles multiples d'estimation est présentée dans la figure 3.7. Elle peut être synthétisée suivant les étapes suivantes :

1. Définir de nombre m de tirage de Monte Carlo.
2. Générer m échantillons aléatoires de chaque facteur d'effort par tirage dans son FDP.
3. Calculer en utilisant chacun des modèles du système d'estimation d'effort les m valeurs de l'effort estimé du nouveau projet étudié.
4. Stocker les $m \times B$ estimations d'effort obtenues dans une structure de données, notamment un tableau.

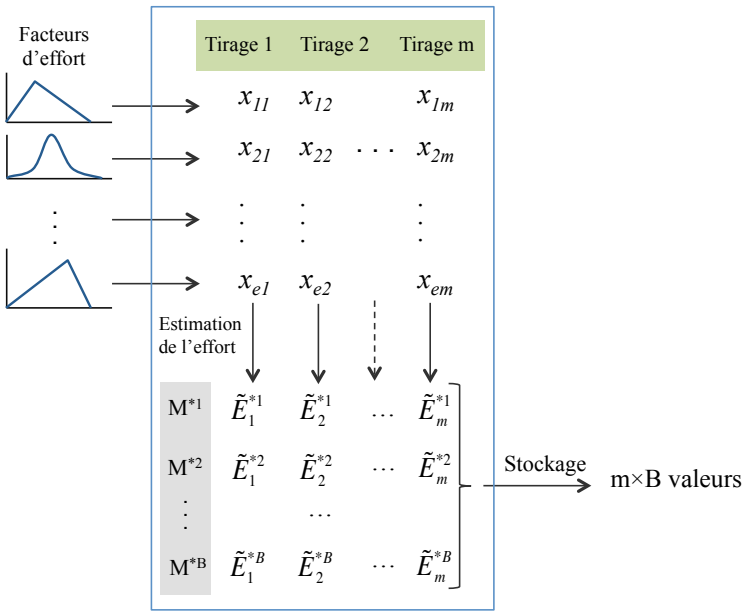


FIGURE 3.7 – Propagation des incertitudes des facteurs d'effort

En raison de l'utilisation d'un système d'estimation constitué de B modèles d'estimation calibré, $m \times B$ itérations sont effectuées dans la simulation de Monte carlo. Le choix d'un large m (de l'ordre de 5000) impliquera un temps de calcul très élevé.

L'algorithme 8 présente et synthétise la procédure de simulation de Monte Carlo appliqué sur un système d'estimation d'effort.

3.2.3 Analyse des résultats de la simulation pour exprimer l'effort estimé

Pour un nouveau projet à estimer, la simulation de Monte Carlo appliquée sur un système d'estimation fourni un ensemble de $m \times B$ estimations de l'effort. Ces estimations peuvent être analysées et étudiées afin d'en tirer des conclusions sur l'effort estimé final du projet étudié. Du point de vue pratique, cet effort estimé peut être exprimé sous la forme d'IvP associé à un degré de confiance. La méthodologie de calcul de l'IvP de l'effort prenant compte des incertitudes du nouveau projet à partir des résultats de simulation Monte Carlo est présentée dans la figure 3.8, elle consiste en 3 étapes :

1. Représentation des FDPs de l'effort estimé à partir des résultats obtenus des modèles du système d'estimation,
2. Calcul des IvPs de l'effort à partir des FDPs obtenues,
3. Calcul de l'IvP final de l'effort du nouveau projet.

Algorithme 8 : Algorithme de Monte Carlo appliqué sur un système d'estimation d'effort pour l'estimation de nouveaux projets dans un environnement incertain

Entrée : Système d'estimation d'effort $\mathscr{S}$
FDP des facteurs d'effort du nouveau projet à estimer
Sortie :Tableau des résultats de la simulation de Monte Carlo
pour $r \in \{1, \cdots, m\}$ **faire**
  pour $i \in \{1, \cdots, e\}$ **faire**
    Génération aléatoire d'un échantillon x_{ir} du $i^{ème}$ facteurs d'effort;
pour $b \in \{1, \cdots, B\}$ **faire**
  pour $r \in \{1, \cdots, m\}$ **faire**
    Estimation d'une valeur d'effort en utilisant le modèle M^{*b} de $\mathscr{S}$ sur $(x_{1r}, x_{2r}, \cdots, x_{er})$;
    Stockage de $\widetilde{E}_r^{*b}$ dans le tableau des résultats
Tableau de $m \times B$ valeurs de l'effort estimé

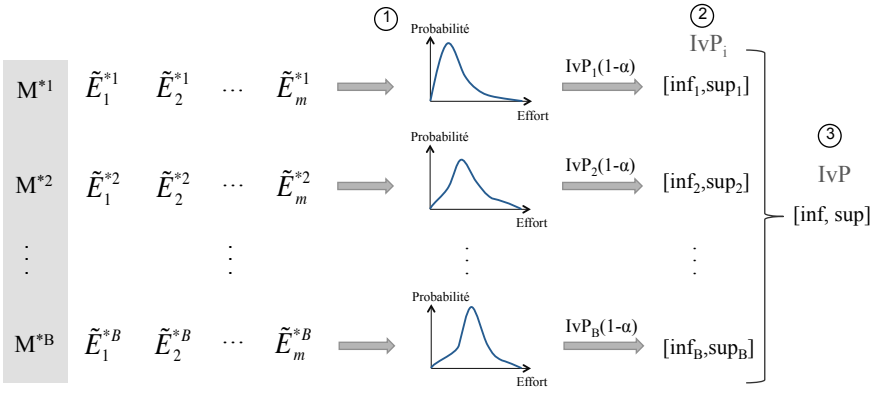


FIGURE 3.8 – Méthodologie de calcul de l'IvP de l'effort à partir des résultats de simulation Monte Carlo

3.2.3.1 Étape 1 : Représentation des FDPs de l'effort estimé par les modèles du système d'estimation

Cette étape commence, tout d'abord par la représentation des estimations d'effort $\widetilde{E}_r^{*b}$ ($r \in \{1, \cdots, m\}$) obtenues de chaque modèle $b \in \{1, \cdots, B\}$ sous la forme d'histogrammes. L'histogramme est une représentation graphique permettant de visualiser la répartition d'une variable. La fonction constituée par les sommets des rectangles de l'histogramme est considérée comme une approximation de la FDP de l'effort estimé. Cependant, en raison de sa forme en escalier, cette fonction n'est pas une approximation satisfaisante de la FDP qui est en général une fonction continue.

Une meilleure approximation de la FDP de l'effort estimé peut être obtenue en utilisant le "polygone de fréquence". Il s'agit de la ligne brisée qui relie les milieux des sommets des rectangles et prolongée de part et d'autre des bornes de l'histogramme de sorte que l'aire sous cette ligne soit égale à 1. Pour obtenir une courbe plus lisse à partir de ce polygone, des méthodes de lissage et d'ajustement peuvent être utilisées telles que les méthodes de noyau et les méthodes par splines [MacCallum+1986].

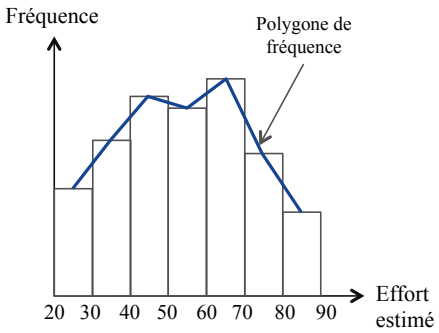


FIGURE 3.9 – Histogramme et polygone des fréquences d'un ensemble de valeurs

3.2.3.2 Étape 2 : Calcul des IvPs de l'effort à partir des FDPs obtenues

Nous visons ici à calculer l'IvP, associé à un degré de confiance, de l'effort à partir de chaque FDP de modèle du système d'estimation. Étant donnée que le système d'estimation est composé de B modèle, B IvPs d'effort seront calculés. L'ensemble des IvPs de l'effort d'un nouveau projet construit à partir des modèles d'un système d'estimation seront noté $\{\text{IvP}_b, b \in \{1, \cdots, B\}\}$. Il n'existe généralement pas de méthode standard de calcul d'IvP, il repose souvent sur des hypothèses et des approximations.

L'observation de la forme de chaque FDP de l'effort estimé obtenue de l'étape précédente permet de distinguer deux cas : (a) la FDP de l'effort estimé se rapproche de la lois normale, (b) la FDP de l'effort estimé n'est pas normale. Dans les deux cas, une vérification de l'hypothèse de normalité est nécessaire. Plusieurs tests statistiques peuvent être utilisés pour cette fin, notamment le test de Shapiro-Wilk [Razali+2011 ; Shapiro+1965].

Nous proposons dans la suite une démarche de calcul d'effort probabiliste pour chacun des cas.

(a) FDP de l'effort estimé se rapprochant de la loi normale

Pour une meilleure approximation de la loi normale, la FDP de l'effort estimé peut être ajustée en utilisant des transformations et par détection et traitement des valeurs aberrantes. Cette FDP peut être exploitée de différentes manières pour exprimer l'effort estimé probabiliste représentant les incertitudes.

L'IvP associé à un degré de confiance $(1-\alpha)$ de l'effort estimé du nouveau projet étudié peut être décrit par l'équation 3.2.8, avec $\widetilde{E}_m$ et s sont respectivement la moyenne et l'écart type des valeurs de l'effort estimé $(\widetilde{E}_r^{*b}), r \in \{1, \cdots, m\}, b \in \{1, \cdots, B\}$ résultant de la simulation de Monte Carlo.

$$IvP(1-\alpha) = \left[\widetilde{E}_m - t_{1-\alpha/2}\, s\sqrt{1+\frac{1}{m\times B}},\quad \widetilde{E}_m + t_{1-\alpha/2}\, s\sqrt{1+\frac{1}{m\times B}}\right] \tag{3.2.8}$$

(b) FDP de l'effort estimé ne suit pas une loi normale

La FDP de l'effort estimé peut être asymétrique et peut présenter plusieurs sommets (FDP multimodale). Il faut tout d'abord analyser les causes de cette forme. Une des raisons peut être liée à l'utilisation d'une FDP discrète d'un facteur d'effort incertain. Ainsi, l'utilisateur peut choisir de réviser et d'ajuster les FDPs des facteurs d'effort ou de traiter et d'analyser la FDPs de l'effort estimé sans apporter de modification. Ainsi, il existe trois options :

1. explorer la possibilité de modifier les facteurs d'effort incertains représentés par des FDPs discrets en des facteurs d'effort certains afin d'obtenir une FDPs normale de l'effort estimé.
2. ou dans le cas d'une FDP d'effort estimé qui a la forme multimodale, séparer les cas afin de la décomposer en plusieurs FDP qui se rapprochent à la lois normales. Par exemple, pour

un facteur d'effort booléen "T.3GL", cette approche consiste à exécuter la simulation Monte Carlo séparément pour le cas ou T.3GL = 0 et pour le cas ou T.3GL = 1. Ainsi, nous obtenons deux FDPs de l'effort estimé. Ces FDP pouvant être approchées par des lois normales, les IvPs ou l'estimation en trois point de l'effort peuvent être générés comme décrit dans (a).

3. ou appliquer le bootstrap sur la FDP non normale de l'effort estimé pour estimer l'effort moyen ainsi que son IvP, de la manière indiquée dans la section 1.3.3.3 du chapitre 1).

3.2.3.3 Étape 3 : Calcul de l'IvP final de l'effort du nouveau projet

Cette étape consiste à dériver l'IvP final de l'effort du nouveau projet à partir des IvPs de l'effort obtenus de l'étape précédente. Afin de prendre en compte les incertitudes caractérisées, l'IvP final de l'effort du nouveau projet doit contenir tous les efforts possibles, en d'autres termes, il doit contenir les IvPs obtenues de l'étape 2 (voir figure 3.10). Ainsi, nous préconisons de prendre comme limite inférieure et limite supérieure de cette IvP respectivement, la limite inférieure minimale et la limite supérieure maximale des IvPs obtenus des modèles du système d'estimation (voir equations 3.2.9).

$$\inf(\texttt{IvP}) = \min_{b\in\{1,\cdots,B\}}\{\inf(\texttt{IvP}_b)\} \quad ; \quad \sup(\texttt{IvP}) = \max_{b\in\{1,\cdots,B\}}\{\sup(\texttt{IvP}_b)\} \tag{3.2.9}$$

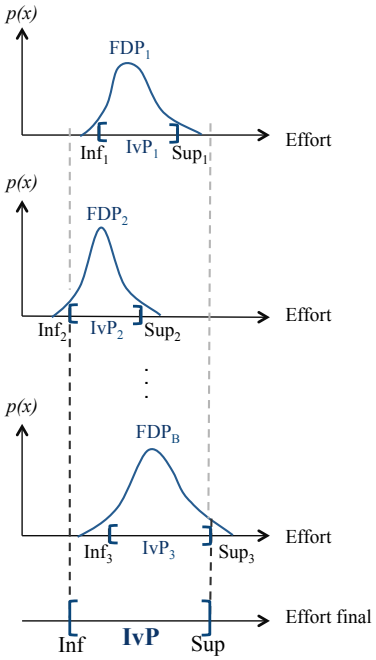


FIGURE 3.10 – Calcul de l'IvP de l'effort du nouveau projet à partir des IvPs obtenus

Conclusion

Estimer les projets de développement logiciel d'une manière performante et réaliste est un défi permanent pour les chercheurs, les praticiens et les organisations de développement.

Dans ce chapitre, deux techniques statistiques sont proposées pour relever ce défi : le bootstrap et Monte Carlo. Le bootstrap est utilisé dans la construction du système d'estimation d'effort réaliste à partir du modèle d'estimation sélectionné. Quand à la méthode de Monte Carlo, elle est exploitée pour représenter les incertitudes relatives aux nouveaux projets dans l'effort estimé.

Les deux techniques ont été sélectionnées pour différentes raisons. Tout d'abord, elles ont une grande applicabilité dans la mesure où elles ne font pas d'hypothèses sur la distribution de la population de projets de développement logiciel représentée par la base de données. De surcroit, elles se caractérisent la facilité de leurs implémentation et programmation en comparaison avec d'autres méthodes telles que l'inférence "Bayésienne".

Afin de construire un système d'estimation d'effort réaliste et le mieux adapté à l'entreprise, le modèle d'estimation sélectionné (par l'approche proposée dans le chapitre précédent) est calibré sur la base de données de l'entreprise en utilisant la technique de bootstrap. Ce couplage du modèle d'estimation avec le bootstrap permet d'intégrer et de représenter les incertitudes du modèle et de la base de données dans le système d'estimation résultant. Ce système d'estimation consiste en plusieurs modèles d'estimation calibrés sur des échantillons de bootstrap tirés de la base de données. Sa performance peut être considérée comme étant similaire à celle du modèle d'estimation à partir duquel il a été construit.

Le système d'estimation d'effort construit permet d'obtenir des estimations d'effort probabilistes reflétant les incertitudes présentes dans la phase de construction. Cependant, il ne reflètent pas les incertitudes relatives aux spécifications des nouveaux projets à estimer. Ces incertitudes sont prises en compte dans notre approche en utilisant la méthode d'échantillonnage Monte Carlo.

Dans l'approche proposée, les incertitudes relatives aux facteurs d'effort sont exprimées à travers des FDPs décrivant le nouveau projet à estimer. Les FDPs sont ensuite propagées dans l'effort estimé final à travers le système d'estimation en utilisant Monte Carlo.

Il est important de noter que cette approche est subjective dans la mesure où les FDPs des facteurs d'effort sont évaluées par l'utilisateur en se référant à son expérience. Il est parfois difficile de formaliser une information ou connaissance subjective du projet en termes de FDPs appropriée. Cette subjectivité est d'autant plus importante en phase très en amont du projet et influence, par conséquent, fortement la distribution de l'effort estimé.

Il est important d'affiner l'estimation de l'effort tout au long du cycle de développement du projet. En avançant dans le projet, les facteurs d'effort deviennent de moins en moins incertains. Par conséquent, les FDPs qui y sont associées doivent être révisées afin de permettre une estimation plus précise et moins incertaine représentant ainsi l'état actuel du projet.

CHAPITRE

Expérimentation et mise en œuvre

Introduction

Nous allons nous intéresser dans ce chapitre, dans un premier temps, à l'expérimentation et la validation de l'approche que nous avons proposée pour la sélection de modèle d'estimation. Quatre modèles d'estimation différents sont évalués et comparés sur cinq bases de données comportant des propriétés différentes et issues de différentes sources.

Cette expérimentation permettra de valider les différents aspects de l'approche et de mieux appréhender les différentes procédures utilisées, leurs intérêts ainsi que les difficultés qu'un utilisateur peut rencontrer.

Dans un deuxième temps, nous expérimentons l'approche de construction d'un système d'estimation ainsi que son utilisation pour l'estimation de l'effort d'un nouveau projet. Cette expérimentation est conduite sur le cas d'étude fourni par un de nos partenaires industriels.

Dans un troisième temps, nous présentons l'implémentation d'une application web permettant l'automatisation de notre approche. Nous présentons, tout d'abord, les fonctionnalités, l'architecture technique et les différents composants de l'application web "EstimSet" que nous avons développée. Ensuite, nous illustrons les fonctionnalités développées dans cette application à l'aide d'un cas d'utilisation.

4.1 Expérimentation et interprétation des résultats de la comparaison de modèles d'estimation

4.1.1 Présentation de l'expérimentation

L'expérimentation est conduite sur trois catégories de bases de données : (1) ISAS qui est une grande base de données multi-organisationnelle, (2) Trois bases de données "PROMISE" qui sont mono-organisationnelles, publiques, et communément utilisées dans la littérature pour l'évaluation et la comparaison de modèles d'estimation d'effort et (3) CAPRI qui provient d'un partenaire industriel du projet ProjEstimate.

L'expérimentation vise à :

— Évaluer un ensemble de modèles d'estimation candidats sur chacune des cinq bases de données,

— Comparer les modèles d'estimation afin de sélectionner le modèle le plus adéquat pour chacune des bases de données.

4.1.1.1 Description des bases de données

La base de données ISAS

La première base de données utilisée dans notre expérimentation est la base de données ISAS collectée en 2009. C'est une base de données multi-organisationnelle qui couvre différentes organisations de plus de 25 pays, différents types d'application et plateformes, etc. Elle peut être utilisée pour différents buts tels que l'étude comparative de la performance d'une entreprise par rapport aux autres représentées dans la base de données, la négociation des développements externalisés et l'estimation de l'effort de développement.

La base de données ISAS est une large bases de données qui peut être utilisée dans le domaine de l'estimation de l'effort de développement logiciel. Elle contient 5052 projets de développement logiciels et 117 attributs.

Les trois bases de données PROMISE : Nasa93, Desharnais, Maxwell

La base de données Nasa93, aussi appelée COCOMO NASA 2, contient 93 projets développés entre 1971 et 1987 dans différents centres de la Nasa (National Aeronautics and Space Administration) aux États Unis. Cette base de données est très utilisée par les travaux de recherche dans le domaine de l'estimation de l'effort de projets de développement logiciel. Les projets de la base de données NASA93 sont décrits selon 16 attributs : la taille fonctionnelle exprimée en kloc, 3 attributs sur le produit (rely, data, cplx), 5 attributs sur le matériel (time, stor, virt, turn, vexp), 4 attributs sur le personnel (acap, aexp, pcap, lexp) et 3 attributs sur le projet(modp, tool, sced). La variable dépendante est l'effort en homme-mois.

La base de données Desharnais contient 81 projets commerciaux développés au sein d'un éditeur de logiciels canadien en 1989. Les projets de la base desharnais sont décrits selon 9 attributs qui sont : TeamExp, ManagerExp, YearEnd, Transactions, Entities, PointsAdjust, Envergure, PointsNonAjust et Language. L'effort en homme-heures est la variable dépendante à estimer dans le cas de cette base de données.

La base de données Maxwell contient 62 projets développés par une des plus grande banque commerciale en Finlande de 1986 à 1993. Les projets de la base de données Maxwell sont décrits

selon 24 attributs qui sont : SYear, App, Har, Dba, Ifc, Source, Telonuse, Nlan, T01, T02, T03, T04, T05, T06, T07, T08, T09, T10, T11, T12, T13, T14, T15 et Size. La variable dépendante est l'effort en homme-heure pour cette base de données.

Les attributs de bases de données Nasa93, Desharnais et Maxwell sont décrits dans les annexes B.1, B.2 et B.3, respectivement.

La base de données CAPRI

La base de données CAPRI est collectée par un des partenaires industriels du projet Projestimate en se basant sur son expérience professionnelle. Elle contient 38 projets datant de 2009 décrits selon 51 attributs.

La table 4.1 récapitule les propriétés de chacune des trois bases de données.

TABLE 4.1 – Aperçu des bases de données PROMISE

Base de données	Organisation	Année	# Projets	# Attributs
ISAS	Organisation anonyme	2009	5052	117
NASA93	NASA	1971-1987	93	16
Desharnais	Éditeur canadien	1983-1989	81	9
Maxwell	Banque Commerciale Finlandaise	1986-1993	62	24
CAPRI	Partenaire industriel ProjEstimate	2009	38	51

4.1.1.2 Modèles d'estimation candidats

Nous choisissons quatre modèles d'estimation à comparer sur ces cinq bases de données, ils sont décrits dans la table 4.2.

TABLE 4.2 – Propriétés des modèles d'estimation candidats

Code	Modèle d'estimation	Type	Hyper-paramètres
RLM	Régression linéaire multiple	Paramétrique	Aucun
RNA	Réseaux de neurones artificiels	Non paramétrique	Nombre de neurones dans la couche cachée et Decay.
FAD	Forêt d'arbres décisionnels	Non paramétrique	Nombre de facteurs d'effort utilisés dans chaque nœud.
k-PPV	k-Plus Proches Voisins	Non paramétrique	Nombre k des plus proches projets voisins à considérer dans l'estimation.

4.1.1.3 Paramètres de l'approche

Dans cette expérimentation, les paramètres des techniques utilisées dans l'approche de sélection sont configurés selon les recommandations que nous avons données dans le chapitre 2.

La plupart des praticiens préconise de choisir un nombre k de validation entre : 3, 5 et 10 [Kohavi1995 ; Krstajic+2014]. Ainsi, dans l'évaluation du modèle d'estimation paramétrique RLM par la VCK, nous choisissons un nombre d'itérations $k = 5$. Pour l'évaluation des trois modèles d'estimation non paramétriques en utilisant la VCI, nous fixons le nombre d'itérations de la boucle extérieure k_{ext} également à 5 pour rester en concordance avec l'évaluation du RLM. Quant au

TABLE 4.3 – Description des attributs sélectionnés de la base de données ISAS

Code d'attribut	Nom complet	Description	Type
AFP	Adjusted Function Point	La taille du projet mesurée par l'IFPUG.	Numérique continu
MTS	Max Team Size	La taille de l'équipe de développement.	Numérique discret
DT	Development Type	Le type de développement du projet : nouveau développement ou amélioration.	Qualitatif
DP	Development Platform	La plateforme primaire de développement	Qualitatif
LT	Language Type	Le type du langage de développement utilisé dans le projet : 3GL, 4GL, etc.	Qualitatif
MA	Methodology Acquired	La méthodologie de développement : achetée ou développée en interne.	Qualitatif
AT	Application Type	Le type de l'application : gestion, système d'information, etc.	Qualitatif
RL	Resource Level	Un niveau en fonction du nombre de personnes dont le temps est inclus dans l'effort reporté.	Numérique discret
UM	Used methodology	Utilisation ou non d'une méthodologie de développement.	Qualitatif
OT	Organisation Type	Le type de l'organisation qui a soumis le projet : commerce, industrie, bancaire, etc.	Qualitatif

nombre d'itérations k_{int} de la boucle intérieure, il sera fixé à 10 pour une sélection performante des hyper-paramètres. Le nombre de bootstrap est fixé à 20 dans cette expérimentation.

Dans la suite, ces modèles d'estimation seront comparés sur chacune des cinq bases de données afin de valider notre approche de sélection d'un modèle d'estimation.

4.1.2 Base de données ISAS

4.1.2.1 Préparation de la base de données ISAS

Cette base de données se caractérise par une grande hétérogénéité et un grand nombre de valeurs manquantes.

Pré-nettoyage

Selon la procédure de préparation de données détaillée dans notre approche (section 2.3.1) et les recommandations du groupe ISAS, nous effectuons les tâches de pré-nettoyage suivantes :

— Les projets dont les informations collectées sont qualifiées de qualité 'C' ou 'D' sont supprimées. Uniquement les projets appartenant à la catégorie 'A' et 'B' sont gardés dans la base de données.

— Les projets dans lesquels la méthode de mesure de la taille fonctionnelle ajustée (AFP) utilisée est différente de l'IFPUG, sont supprimés.

— Les attributs constants et les projets dont l'effort renseigné est null sont supprimés.

— S'il existe deux ou plusieurs attributs corrélés entre eux, il faudra garder l'attribut le plus significatif pour l'estimation de l'effort, par exemple dans l'ISAS la taille fonctionnelle (FP) et la taille fonctionnelle ajustée (AFP).

Après l'étape de pré-nettoyage, la base de données résultante contient 1252 projets et 9 attributs, en plus de l'effort, qui sont : MTS, DT, DP, LT, MA, AT, RL, UM et OT. Ces 9 attributs ainsi que l'attribut de la taille des projets sont décrits dans la table 4.3.

Sélection des facteurs d'effort

Nous effectuons des tests de corrélation des attributs avec l'effort afin de sélectionner les facteurs d'effort parmi les 9 attributs. La table 4.4 présente les résultats du test de Spearman sur les attributs numériques et la table 4.5 présente les résultats de F-Test ANOVA sur les attributs qualitatifs (nominal). Les coefficients *rho* et F de Spearman et ANOVA, respectivement, confirment que la taille AFP est le facteur d'effort le plus important, et montrent que 7 attributs sont corrélés avec l'effort : AFP, MTS, DT, UM, DP, LT, OT et MA. Ces attributs sont retenus dans cette étape de pré-sélection. Ensuite, l'importance de chacun des attributs retenus est déterminée par la méthode de filtrage basée sur les forêts d'arbres décisionnels (voir section 2.3.1.3). Les résultats sont présentés dans la table 4.6. Nous retenons les attributs ayant une grande importance (supérieure à 10), ainsi, les attributs sélectionnés comme facteurs d'effort sont : AFP, MTS, DP, LT, DT et UM.

TABLE 4.4 – Coefficients de corrélation Spearman sur les attributs quantitatifs de l'ISAS (sig. : significatif)

Attribut	Coefficient de correlation Spearman (sig.\non sig.)
AFP	0.658 (sig.)
MTS	0.567 (sig.)
RL	0.1 (non sig.)

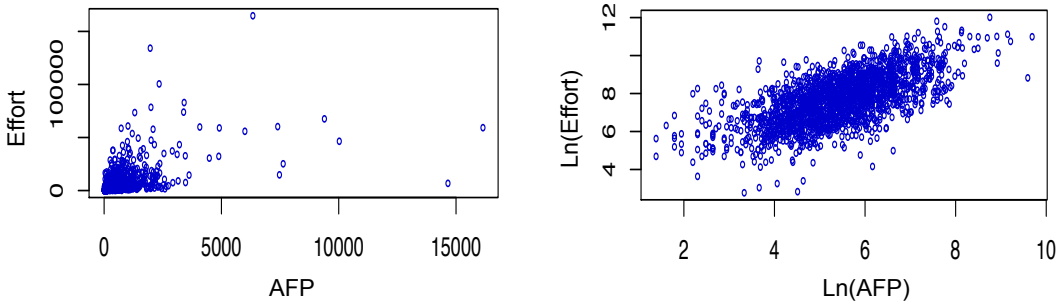
TABLE 4.5 – F-test ANOVA sur les attributs qualitatives de l'ISAS

Attribut	F-test ANOVA (sig.\non sig.)	P-value
DT	51.688	0.0002 (sig.)
UM	7.624	0.0061 (sig.)
DP	9.67	0.0096 (sig.)
LT	4.950	0.012 (sig.)
OT	2.76	0.0002 (sig.)
MA	1.278	0.0016 (sig.)
AT	1.476	0.25 (non sig.)

TABLE 4.6 – Importances des attributs pré-sélectionnés de l'ISAS

Attribut	Importance
AFP	69.25
MTS	52.61
DP	25.05
LT	21.01
DT	17.79
UM	16.78
MA	9.278
OT	8.13

Comme déjà mentionné dans le chapitre 2, la relation entre la taille de projet et l'effort n'est pas linéaire. Ce constat de non linéarité est illustré sur la figure 4.1a pour le cas de la base de données ISAS. Plusieurs travaux de recherche supposent que la fonction décrivant la relation entre l'effort et la taille a une forme exponentielle. La figure 4.1b confirme cette hypothèse pour le cas de la base de données ISAS.



(a) Effort en fonction de la taille AFP

(b) ln(Effort) en fonction de ln(AFP)

FIGURE 4.1 – Linéarisation de la relation entre l'effort et la taille (AFP) par l'application du logarithme

Les attributs non significatifs pour l'estimation de l'effort ainsi que les projets avec des valeurs manquantes dans les facteurs d'effort sont supprimés de la base de données. La base de données résultante à ce stade est complète et contient 548 projets.

Détection et traitement des projets aberrants

Pour la détection et la suppression des projets aberrants, nous utilisons le tracé de chi deux (X^2) qui se base sur la distance de Mahalanobis D^2. C'est un outil graphique permettant de visualiser les projets aberrants (voir figure 4.2). Il trace les distances robustes Mahalanobis des projets de la base de données, ordonnées de la plus petite à la plus grande par rapport aux quantiles de la distribution de X^2. Si les données suivent la loi normale, les quantiles en ordonnées et les distances des projets en abscisses doivent se correspondre et former une ligne droite. Ainsi, les projets aberrants peuvent être détectés visuellement. Ils sont les projets avec les plus grandes distances D^2.

La figure 4.2 présente le tracé de chi deux pour la base de données ISAS. Ce tracé montre qu'il existe approximativement 20 projets aberrants. Après suppression de projets aberrants, la base de données préparée finale est composée de 518 projets.

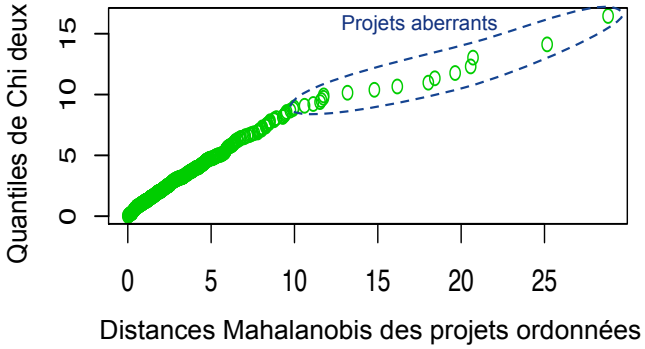


FIGURE 4.2 – Tracé de chi deux en fonction des distances D^2 ordonnées des projets de l'ISAS

4.1.2.2 Évaluation des modèles d'estimation candidats sur l'ISAS et analyse des résultats

La table 4.7 présente les résultats de l'évaluation des modèles d'estimation candidats sur la base de données ISAS après préparation. Nous remarquons que le modèle d'estimation RLM présente les

meilleurs (plus faibles) *MMRE* (=55.61%) et *MdMRE* (=39.62), et le meilleur (plus grand) *Pred*(0.25) (=33.21). Il présente donc la meilleure performance prédictive. De surcroit, il présente le plus faible *MRMD* (=11.24), ce qui indique qu'il est le modèle d'estimation d'effort le moins incertain parmi les modèles d'estimation candidats pour le cas de la base de données ISAS. Ce modèle montre une légère tendance à sous-estimer l'effort.

Les modèles d'estimation RNA et FAD montrent des résultats similaires. Le modèle RNA dépasse légèrement FAD dans la performance prédictive, cependant, le FAD est légèrement moins incertain que RNA. Le modèle k-PPV est classé le dernier sur tous les indicateurs de performance. Sa performance prédictive est acceptable, Cependant, il est le modèle d'estimation d'effort le plus incertain (*MRMD*=32.16).

TABLE 4.7 – Performances des modèles d'estimation candidats sur la base de données ISAS

Modèle d'estimation	% *MMRE*	% *MdMRE*	% *Pred*(0.25)	% *MRMD*	% *Und*(0.25)	% *Ovr*(0.25)
RLM	**55.61**	**39.62**	**33.21**	**11.24**	17.76	15.45
RNA	60.50	41.58	31.29	28.31	17.76	13.53
FAD	57.16	42.14	29.92	26.01	17.57	12.35
k-PPV	65.15	44.72	27.02	32.16	16.4	10.63

Indicateur MRMD et précision des estimations

Pour illustrer et bien appréhender la relation entre le facteur *MRMD* et la précision des estimations, nous représentons pour l'ensemble des projets de la base de données la distribution des estimations multiples d'effort dans les figures 4.3. Ces estimations sont produites en utilisant le bootstrap et sont ordonnées selon la moyenne des estimations pour chaque projet. La moyenne est représentée dans les figures par une ligne qui passe au milieu des points. Les figures montrent et confirment que plus le *MRMD* est grand, plus la précision des estimations est faible.

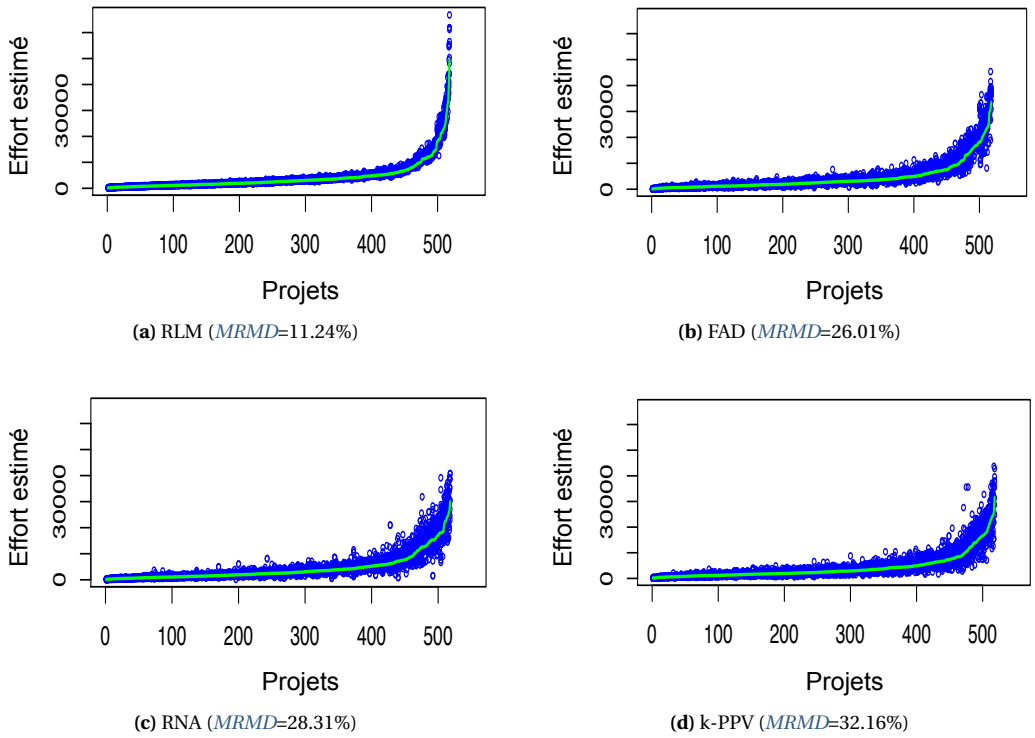
Facteurs d'effort sélectionnés et performance de modèle d'estimation

Pour mettre en exergue l'importance de la bonne sélection de facteurs d'effort, nous avons expérimenté le modèle d'estimation RLM avec trois différents ensembles de facteurs d'effort (F1, F2 et F3). Les résultats de l'expérimentation sont décrits dans la table 4.8, ils indiquent que F3 présente les *MMRE* (71%) et *MdMRE* (51.24%) les moins bons. F1 montre également des *MMRE* et *MdMRE* insatisfaisants par rapport au F2.

Ainsi, cette expérimentation montre qu'intégrer de nombreux facteurs d'estimation peut dégrader la qualité du modèle d'estimation surtout quand il s'agit d'une base de données de petite ou moyenne taille. Le grand nombre de facteurs d'effort complexifie le modèle d'estimation et rend l'apprentissage difficile et nécessite plus d'informations. De même, la minimisation du nombre de facteurs d'effort conduit à une perte d'informations utiles et importantes pour l'estimation. Certains facteurs d'effort sont importants pour améliorer la performance du modèle d'estimation construit, même s'ils n'influencent pas fortement l'effort.

TABLE 4.8 – Performance du modèle d'estimation RLM en fonction des facteurs d'effort sélectionnés

Ensemble	Facteurs d'effort	% *MMRE*	% *MdMRE*
F1	AFP, MTS, DT, DP, LT, MA, AT, RL, UM, OT	65.94	43.45
F2	AFP, MTS, DT, UM, DP et LT	55.61	39.62
F3	AFP, MTS, DT, UM	71	51.24



(a) RLM (*MRMD*=11.24%)

(b) FAD (*MRMD*=26.01%)

(c) RNA (*MRMD*=28.31%)

(d) k-PPV (*MRMD*=32.16%)

FIGURE 4.3 – Les résultats du bootstrap sur l'ISAS

4.1.3 Bases de données PROMISE

Nous appliquons notre approche sur les trois bases de données "PROMISE" : Nasa93, Desharnais et Maxwell.

4.1.3.1 Préparation des bases de données PROMISE

Pré-nettoyage

Parmi les trois bases de données PROMISE, uniquement Desharnais présente des valeurs manquantes. En effet, quatre projets (38,44,66,75) contiennent des données incomplètes, Nous les avons donc supprimés comme la majorité des travaux de recherche.

Sélection des facteurs d'effort

Dans le cas de la base de données Nasa93, nous considérons les 15 attributs en plus de la taille fonctionnelle en kloc comme étant des facteurs d'effort, comme dans le modèle COCOMO2. Pour la bases de données Desharnais, les tests de corrélation de Spearman et ANOVA ont montré l'importante influence de 5 attributs : PointsNonAjust, Transactions, Entities, Envergure, Language. Quant à la base de donnée Maxwell, les tests ont montré l'importance de 9 attributs : Size, SYear, T01, T07, T08, T09, T10, T11, T14.

Détection et traitement des projets aberrants

La procédure de détection des projets aberrants est conduite de la même manière que pour l'ISAS en utilisant le tracé de chi deux en fonction des distances D^2. Trois projets aberrants ont été détectés dans Nasa93, deux dans Desharnais et un seul dans Maxwell. Nous avons supprimé ces

projets aberrants des bases de données. La table 4.9 présente les résultats de sélection de facteurs d'effort ainsi que les projets aberrants détectés pour les trois bases de données PROMISE.

TABLE 4.9 – Résultats des préparations des bases de données PROMISE

Base de données	Facteurs d'effort	Projets aberrants
Nasa93	rely, data, cplx, time, stor, virt, turn, vexp, acap, aexp, pcap, lexp, modp, tool, sced (voir annexe B.1)	59, 98, 91
Desharnais	PointsNonAjust, Transactions, Entities, Envergure, Language (voir annexe B.2)	29
Maxwell	Size, SYear, T01, T07, T08, T09, T10, T11, T14 (voir annexe B.3)	26

4.1.3.2 Évaluation des modèles d'estimation candidats sur les bases de données PROMISE et analyse des résultats

La table 4.10 présente les résultats de l'évaluation des modèles d'estimation sur les base de données PROMISE.

Dans le cas de la base de données Nasa93, le modèle d'estimation RNA montre les meilleurs indicateurs de performance prédictive avec un *MMRE* de 55.66%, un *MdMRE* de 30.96% et un *Pred*(0.25) de 38.37%. Pour l'indicateur d'incertitude, le RNA montre un *MRMD* très légèrement inférieur à celui du RLM, toutefois, le RNA reste plus performant que le RLM.

La base de données Desharnais présente résultats différents. Le modèle d'estimation FAD montre les meilleurs performances avec un *MMRE* de 58%, un *MdMRE* d'environ 42%, un *Pred*(0.25) de 38% et un *MRMD* de 19.28%. Le k-PPV présente des performances très proches avec un *MMRE* supérieur à celui du FAD, ce qui montre qu'il peut présenter un risque supérieur de produire de mauvaises estimation.

Sur la base de données Maxwell, le modèle d'estimation RLM montre un meilleur *MMRE* d'environ 18% mais pas les meilleurs *MdMRE* et *Pred*. Le modèle RNA montre des performances proches mais de meilleurs *MdMRE* et *Pred*(0.25). Cependant, le RLM est le moins incertain et le plus précis en mode probabiliste. Il montre, en effet, le plus petit *MRMD* qui est d'environ 18%. Ainsi, le RLM est le modèle le plus adéquat pour cette base de données selon notre approche qui préconise l'estimation en mode probabiliste.

TABLE 4.10 – Performances des modèles d'estimation candidats sur les bases de données PROMISE

Base de données	Modèle d'estimation	% *MMRE*	% *MdMRE*	% *Pred*(0.25)	%*MRMD*
Nasa93	RLM	56.52	36.97	36.43	11.93
	RNA	55.66	30.96	38.37	12.38
	FAD	58.98	39.89	32.28	28.84
	k-PPV	62.64	39.84	36.49	29.08
Desharnais	RLM	65.1	44.54	33.38	21.72
	RNA	97.11	55.42	29.56	44.36
	FAD	58.05	41.98	38.01	19.28
	k-PPV	68.11	42.91	38.24	23.69
Maxwell	RLM	48.61	34.29	33.85	17.93
	RNA	51.11	32.85	37.18	47.94
	FDA	58.8	39.87	25.64	24.47
	k-PPV	73.32	40.53	25.9	31.63

Pour récapituler, le modèle d'estimation le plus performant :

— pour Nasa93 : RLM,

— pour Desharnais : FAD,

— pour Maxwell : RLM.

4.1.4 Base de données CAPRI

4.1.4.1 Préparation de la base de données CAPRI

Pré-nettoyage

La base de données CAPRI est complète, elle ne présente pas de valeurs manquantes.

Parmi les 51 attributs analysés de la base de données, nous avons sélectionné 8 attributs comme étant potentiellement déterminants dans l'estimation de l'effort. Ces attributs sont : PF total, service, direction, Nb applications, Nb services, nature, type et technologie. Ils sont décrit dans la table 4.11.

TABLE 4.11 – Description des facteurs d'effort retenus de la base de données CAPRI

Attribut	Description	Type
PF total	La mesure de la taille fonctionnelle totale du projet en PF.	Numérique
Direction	La direction de développement porteur du projet et chargée de sa gestion et pilotage, example : dans la banque les directions sont : crédit, bource, RH, etc.	Qualitatif
Service	Le service responsable sur le projet, exp : analyste métier, architecture, assurance qualité, etc.	Qualitatif
Nb applications	Le nombre d'applications impactées par le projet et doivent être modifiées afin de s'y adapter.	Numérique
Nb services	Nombre de services contributeurs au projet.	Numérique
Nature	La nature du projet : nouveau ou évolution	Qualitatif
Type	Type du projet : développement, intégration progiciel ou combinaison.	Qualitatif
Technologie	Technologie de développement : mainframe, web, composite ou client/serveur.	Qualitatif

Sélection des facteurs d'effort

Il est important de noter qu'en raison de la petite taille de la base de données CAPRI, le nombre de facteurs d'effort à sélectionner est limité. Ainsi, il est important d'optimiser (minimiser) le nombre de facteurs d'effort sélectionnés dans l'étape de sélection de facteurs d'effort (voir section 2.3.1.3). L'analyse de corrélation de Spearman montre qu'en plus de l'attribut relatif à la taille du projet (PF total), les attributs Nb d'applications et Nb services sont corrélés avec l'effort. Ainsi en plus de la taille de projet, ces deux attributs sont retenus. L'ANOVA montre que 3 parmi les 6 attributs qualitatifs sont déterminants de l'effort, ils sont : nature, type et technologie. Les résultats des tests Spearman et ANOVA sont présentés dans les tables 4.12 et 4.13.

TABLE 4.12 – Coefficients de corrélation Spearman sur les attributs quantitatifs de CAPRI (sig. : significatif)

Attribut	Coefficient de correlation Spearman (sig.\non sig.)
PF total	0.88 (sig.)
Nb applications	0.57 (sig.)
Nb services	0.44 (sig.)

TABLE 4.13 – Coefficients F du test ANOVA sur les attributs qualitatifs de CAPRI

Attribut	F-test ANOVA (sig.\non sig.)
Direction	0.038(non sig.)
Service	0.96 (non sig.)
Nature	4.68 (sig.)
Type	1.76 (sig.)
Technologie	2.21 (sig.)

Détection et traitement des projets aberrants

Le tracé de chi deux pour la détection des projets aberrants montre l'existence de deux projets aberrants pour dans la base de données CAPRI (voir figure 4.4). Après la suppression de ces projets aberrants, la base de données résultante et finale contient 36 projets.

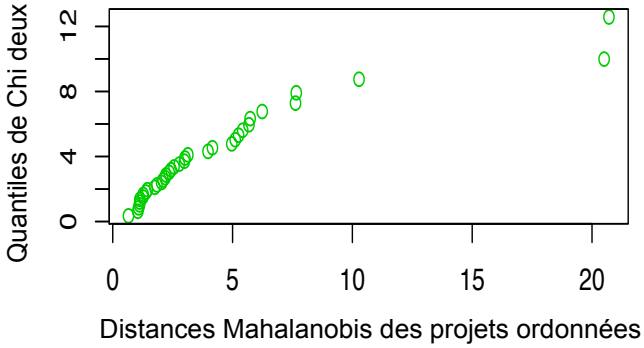


FIGURE 4.4 – Tracé de chi deux en fonction des distances D^2 ordonnées des projets de CAPRI

4.1.4.2 Évaluation des modèles d'estimation candidats sur la base de données CAPRI et analyse des résultats

Les résultats de l'évaluation des modèles d'estimation candidats sur la base de données CAPRI sont présentés dans la table 4.14. Nous constatons que le modèle d'estimation RLM montre les meilleurs indicateurs *MMRE* (=46.01%) et *Pred*(0.25) (=31.33), il est donc le plus adéquat pour une stratégie d'aversion pour le risque.

Nous remarquons aussi que le modèle d'estimation FAD présente les meilleurs *MdMRE* (= 41.46) et *MRMD* (=20.91%). Ainsi, il est le moins incertain et le plus précis.

TABLE 4.14 – Performances des modèles d'estimation candidats sur la base de données CAPRI

Modèle d'estimation	% *MMRE*	% *MdMRE*	% *Pred*(0.25)	% *MRMD*	% *Und*(0.25)	% *Ovr*(0.25)
RLM	**46.01**	43.28	**31.33**	24.78	14.0	17.33
RNA	56.52	54.81	25.33	42.2	14.67	10.67
FAD	47.32	**41.46**	28.67	**20.91**	17.33	11.33
k-PPV	55.44	48.22	24.67	22.91	18.0	6.67

Nous présentons le diagramme en boites à moustaches des *MRE* des quatre modèles d'estimation d'effort sur la base de données étudiée afin de mieux appréhender leurs comportements prédictifs (voir figure 4.5). Nous remarquons que les RLM et FAD ont le même nombre de *MRE* aberrants, c'est à dire le même nombre d'estimation d'effort de qualité très faible. Le FAD présente la plus petite boite qui représente la petite variabilité de 50% des *MRE*s par rapport aux autres modèles d'estimation. Cependant, le RLM présente les moins larges extrémités ce qui explique pourquoi la moyenne (*MMRE*) est plus petite pour ce modèle.

En conclusion, sur cette base de données, les modèles d'estimation RLM et FAD ont des performances similaires et comparables avec une tendance de sur-estimation pour RLM et de sous-estimation pour le FAD. Le modèle d'estimation RNA est le plus incertain avec un plus grand *MRMD* de 42.2%.

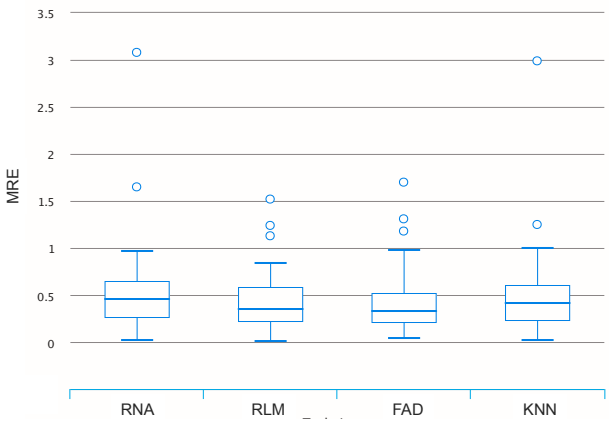
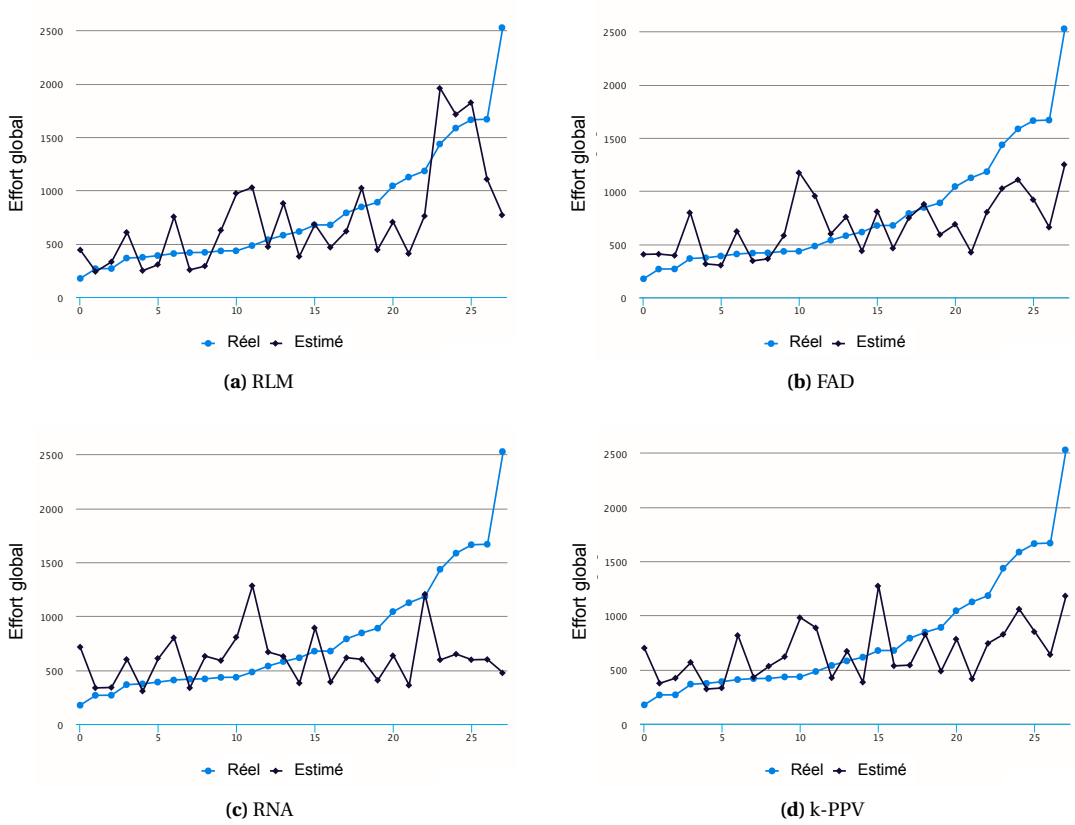


FIGURE 4.5 – Boites à moustaches des modèles d'estimation d'effort sur la base de données CAPRI

Pour avoir une vision complète et claire des comportements des modèles d'estimation sur CAPRI, nous présentons dans la figure 4.6 les efforts réels ordonnées et les efforts estimés y associés, obtenus par chacun des modèles d'estimation candidats. Nous remarquons que le modèle RLM montre des bonnes performances pour les petits et moyens projets mais diverge dans le cas des grands projets. Le modèle FAD suit mieux la tendance de la courbe des efforts réels même pour les grands efforts. Nous constatons également que le RNA diverge pour plusieurs projets de différents ordres de grandeurs, il est le modèle d'estimation le moins performant.



(a) RLM

(b) FAD

(c) RNA

(d) k-PPV

FIGURE 4.6 – Graphes des efforts réels et efforts estimés par les modèles d'estimation candidats pour la base de données CAPRI

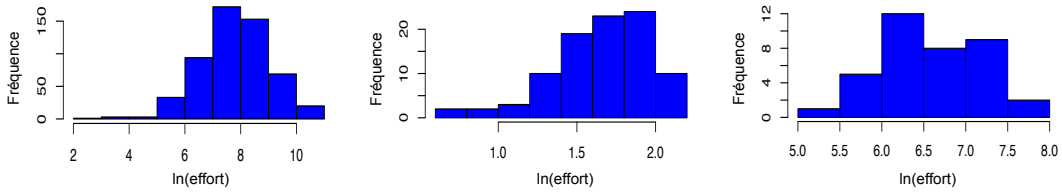
4.1.5 Discussion/Comparaison de modèles d'estimation

La table 4.15 présente une synthèse des modèles d'estimation sélectionnés sur les cinq bases de données. Nous remarquons que le modèle d'estimation sélectionné diffère d'une base de données à l'autre. Ceci confirme qu'il n'existe pas de modèle d'estimation d'effort unique performant sur toute base de données de projets logiciels. En effet chaque base de données présente des spécificités qui font qu'un modèle soit plus adapté par rapport à un autre. Par exemple l'ISAS est multi-organisationnelle, grande et la distribution des efforts de ses projets se rapproche bien de la loi normale (voir figure 4.7). Ainsi le modèle d'estimation RLM qui se base sur l'hypothèse de normalité fournit de bons résultats sur cette base. Contrairement à l'ISAS, les histogrammes des bases de données Nasa93 et CAPRI montrent que les efforts s'éloignent un peu de la loi normale. Ceci explique pourquoi le RLM ne présente pas les meilleurs performances pour ces bases.

Les résultats de l'expérimentation montrent également que la sélection d'un modèle d'estimation adéquat pour une base de données n'est pas toujours une tâche simple en raison de nature multi-dimensionnelle des critères de sélection. Parfois, l'utilisateur peut se confronter à une situation où un modèle d'estimation A est meilleur en terme de performance prédictive et un autre modèle B est meilleurs en terme d'incertitude/précision. Dans ce cas, le choix dépendra des besoins de l'utilisateur. S'il s'agit d'une phase très en amont du projet qui demande une estimation grossière, alors la précision de l'intervalle de l'estimation (largeur) n'est pas plus importante que son exactitude (la probabilité de contenir l'effort réel).

TABLE 4.15 – Synthèse des modèles d'estimation sélectionnés

	ISAS	Nasa93	Desharnais	Maxwell	CAPRI
Modèle d'estimation sélectionné	RLM	RNA	FAD	RLM	FAD
% *MMRE*	55.61	55.66	58.05	48.61	46.42
% *MdMRE*	39.62	30.96	41.98	34.29	37.42
% *Pred*(0.25)	33.21	38.37	38.02	33.85	28.67
% *MRMD*	19.8	12.38	19.28	17.93	21.18



(a) ISAS **(b)** Nasa93 **(c)** CAPRI

FIGURE 4.7 – Histogrammes des ln(effort) des bases de données ISAS, Nasa93 et CAPRI

Nous avons classé, pour chaque base de données, les modèles d'estimation évalués et nous avons calculé les classements moyens sur l'ensemble des bases de données. La table 4.16 des résultats montre que le RLM et FAD surpassent les autres modèles d'estimation pour cet ensemble de bases de données.

TABLE 4.16 – Classements des modèles d'estimation candidats sur les bases de données de l'expérimentation

Modèle de d'estimation	Classement dans la base de données					Moyenne des classements
	ISAS	Nasa93	Desharnais	Maxwell	CAPRI	
RLM	1	2	2	1	2	1.6
RNA	3	1	3	3	4	2.8
FAD	2	3	1	2	1	1.8
k-PPV	4	4	2	4	3	3.4

4.2 Expérimentation de l'approche de construction d'un système d'estimation - Estimation de l'effort d'un nouveau projet

Nous expérimentons, dans cette section, l'approche que nous avons proposée dans le chapitre 3 pour la construction de système d'estimation réaliste et l'utilisation de ce système pour l'estimation d'un nouveau projet. Nous nous intéressons au cas de notre partenaire industriel présenté par la base CAPRI.

4.2.1 Construction d'un système d'estimation pour CAPRI

Pour la base de données de CAPRI, les résultats de comparaison de modèles d'estimation ont montré que les modèles RLM et FAD présentent les meilleures performances.

TABLE 4.17 – Description des variables du système d'estimation construit pour CARPI

Facteur d'effort	Description	Type
PF	PF total	Numérique
NbServ	Nb services	Numérique
NbApp	Nb applications	Numérique
NatNv	Nature de projet : Nouveau	Booléen
TypDev	Type de projet : Développement	Booléen
TypInt	Type de projet : Intégration progiciel	Booléen
TechMain	Technologie de développement : Mainframe	Booléen
TechWeb	Technologie de développement : Web	Booléen

Nous choisissons, dans cette section, de construire le système RLM d'estimation pour CAPRI en se basant sur le modèle RLM pour deux raisons : (1) il donne de bons résultats sur CAPRI et (2) pour des raisons didactiques nous l'avons préféré sur FAD afin de rendre les explications plus faciles, mais la même chose pourrait être faites avec FAD. Nous fixons le nombre de modèles composant le système d'estimation à 10 car la base de données manipulée est de petite taille. Chaque modèle $M^{*b}, b \in 1,\cdots,10$ du système d'estimation peut être décrit comme dans l'équation 4.2.1. Les variables des modèles d'estimation sont décrites dans la table 4.17 et les coefficients des modèles, obtenus après apprentissage sur les échantillons bootstrap de la base de données CAPRI, sont présentés dans la table 4.18.

$$log(\widetilde{E}^{*b}) = \beta_{b0} + \beta_{b1} log(PF) + \beta_{b2} NbServ^{0.198} + \beta_{b3} NbApp^{-0.648} + \beta_{b4} NatNv + \beta_{b5} TypDev + \beta_{b6} TypInt + \beta_{b7} TechMain + \beta_{b8} TechWeb \quad (4.2.1)$$

TABLE 4.18 – Coefficients des modèles composant le système d'estimation de CAPRI

M^{*b}	β_{b0}	β_{b1}	β_{b2}	β_{b3}	β_{b4}	β_{b5}	β_{b6}	β_{b7}	β_{b8}
M^{*1}	1.548	0.596	0.286	0.20	-0.21	0.025	-0.834	-0.37	-0.303
M^{*2}	0.270	0.736	0.174	-0.394	0.236	0.931	-0.097	-0.50	-1.004
⋮					⋱				
M^{*10}	0.425	0.428	0.610	0.533	-0.074	-0.096	-0.133	0.075	-0.032

4.2.2 Estimation de l'effort d'un nouveau projet

Nous conduisons cette expérimentation sur trois projets connus de l'entreprise qui n'étaient pas dans la base de données CAPRI. Les facteurs d'effort ont été identifiés par l'expert d'une façon probabiliste lors du lancement du projet. Les valeurs réelles des projets, telles que la taille fonctionnelle et l'effort, ont été déterminées lorsque les projets sont achevés. Dans cette expérimentation, nous illustrons la procédure d'utilisation du système d'estimation construit pour l'estimation de l'effort d'un nouveau projet présentant des incertitudes. Nous cherchons également à mettre en évidence l'importance de l'évaluation probabiliste des facteurs d'effort et la propagation des incertitudes par Monte Carlo. Pour cela, nous allons comparer l'effort réel du nouveau projet à estimer aux IvPs de l'effort obtenus des deux cas suivants :

(a). Tous les facteurs d'effort sont évalués d'une façon déterministe.

(b). Un ou plusieurs facteurs sont évalués d'une façon probabiliste.

4.2.2.1 Description des nouveaux projets

Le premier projet est de type "Évolution", de nature "Développement" et utilise une technologie "Composite". Ce projet impacte deux applications, et implique 6 services de l'organisation.

L'expert indique que la taille de projet est généralement sous-estimée avec un taux qui varie entre 10 à 50% selon la phase de la réalisation de l'estimation. Pour le nouveau projet à estimer dans cette expérimentation, notre partenaire industriel évalue la taille fonctionnelle à 145 PF. En mode probabiliste pour tenir compte des incertitudes, l'expert décrit la taille fonctionnelle par une FDP triangulaire d'une valeur minimale de 130 PF, une valeur la plus probable de 145 PF et une valeur maximale de 180.

Ce projet ainsi que les deux autres projets sont décrits dans la table 4.19.

TABLE 4.19 – Description des nouveaux projets à estimer

Projet	Type	Nature	Tech	Nb app	Nb serv	Taille fonctionnelle estimé en amont
1	Évolution	Développement	Composite	2	6	Triang(130,145,180)
2	Nouveau	Combinaison	Composite	2	3	Unif(300,400)
3	Nouveau	Développement	Web	3	14	Unif(150,190)

4.2.2.2 Calcul de l'IvP de l'effort des projets

Nous détaillons dans cette section la démarche d'estimation de l'effort du premier projet. La même démarche est suivie pour les deux autres projets.

Pour le cas (a), la distribution de l'effort estimé pour ce premier nouveau projet en utilisant le système d'estimation est présenté dans la figure 4.8. L'IvP de l'effort associé à un degré de confiance de 95% est égal à [576,633] mois-hommes. Cet IvP représente les incertitudes de la base de données et du modèle d'estimation RLM.

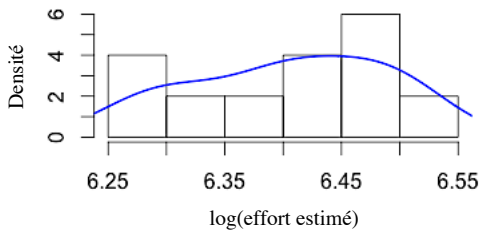


FIGURE 4.8 – Histogramme et FDP du log de l'effort estimé du premier projet (cas (a))

Pour le cas (b), nous avons appliqué la simulation Monte Carlo sur chacun des modèle du système d'estimation afin de propager les incertitudes de la taille du nouveau projet. La figure 4.9 présente les résultats de la simulation. Les estimations d'effort obtenues de chaque modèle sont analysées. Nous remarquons que les modèles du système d'estimation se comportent de façons différentes, les distributions de l'effort estimé ont différentes formes. Les estimations produites par le modèle M^{*1} varient entre 609 et 660 mois-hommes alors qu'elles sont plus importantes pour le modèle M^{*2} et varient entre 679 et 750 mois-hommes. L'IvP de l'effort de chaque modèle est ensuite calculé pour un degré de confiance fixé à 95%. La limite inférieur minimale correspond au modèle M^{*6} et égale à 555 mois-hommes, alors que la limite supérieure maximale correspond au modèle M^{*2} et égale à 716 mois-hommes. Ainsi, l'IvP finale de l'effort du nouveau projet est [555,716].

Nous remarquons que l'IvP obtenu en utilisant la simulation Monte Carlo (cas (b)) comporte l'effort réel. L'IvP de l'effort obtenu sans tenir compte de incertitudes des facteurs d'effort (cas (a))

ne comporte pas l'effort réel. Ainsi, même si l'expert s'appuie sur la limite max de ce dernier (633 mois-hommes), il aura un dépassement d'effort à la fin du projet qui est égal à 44 mois-hommes (677-633). La table 4.20 récapitule les résultats de l'estimation de ce premier projet ainsi que les résultats des deux autres projets.

Pour le deuxième projet, les résultats montrent que l'IvP obtenu du cas (a) ne contient pas l'effort réel et que la limite inférieure de cet IvP dépasse légèrement l'effort réel. Il s'agit d'une sur-estimation. Ainsi, l'utilisation de cet IvP n'entrainera pas de dépassement de coûts. L'IvP obtenu par la simulation Monte Carlo (cas (b)) contient l'effort réel mais n'est pas centré sur cet effort. Pour ce projet, l'intégration des incertitudes des facteurs d'effort n'améliore pas considérablement la précision de l'IvP.

Les résultats du troisième projet, comme pour le premier projet, montre que l'IvP déterminé dans le cas (a) ne contient pas l'effort réel et son utilisation peut entrainer un risque de dépassement de coûts. Même avec l'utilisation de la limite supérieure de cet IvP, il y a un dépassement de 19 mois-hommes.

Cette expérimentation montre, pour le cas de deux sur trois projets, l'intérêt de prendre en compte, dans l'estimation de l'effort, les incertitudes des nouveaux projets représentées dans les facteurs d'effort. Ces incertitudes sont aussi importantes que les deux types d'incertitudes intégrés dans le système d'estimation.

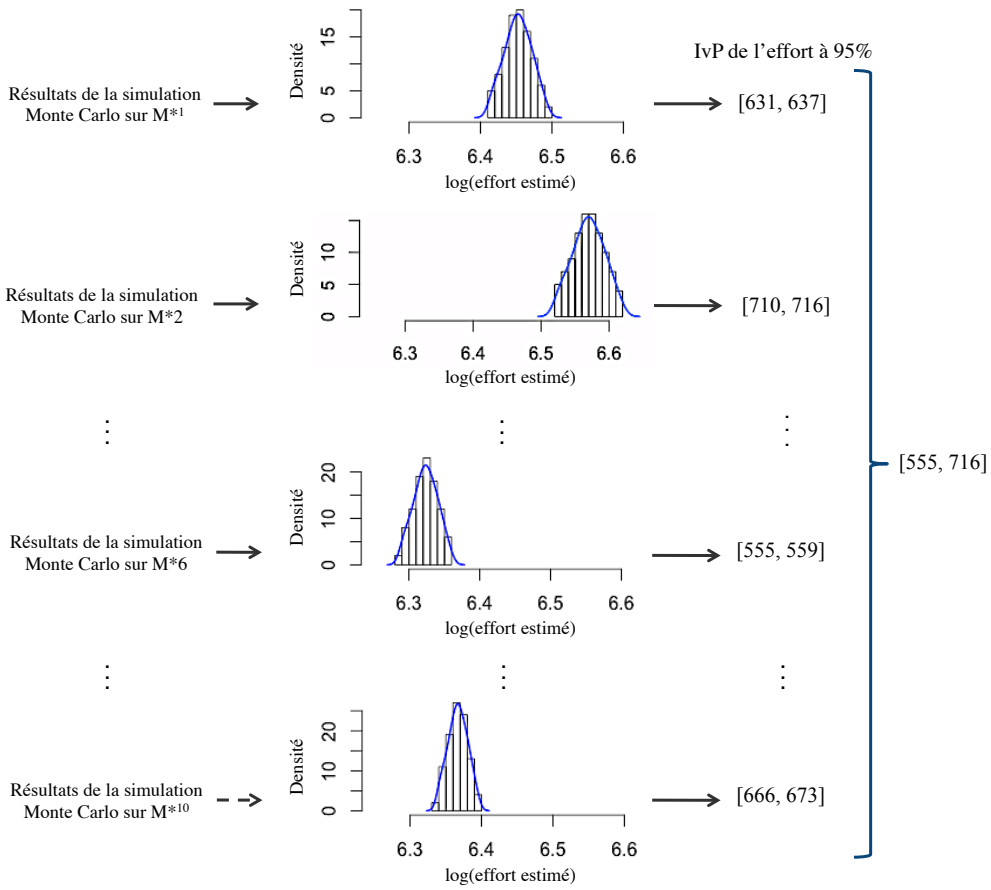


FIGURE 4.9 – Calcul de l'IvP de l'effort du premier projet en utilisant la simulation Monte Carlo (cas (b))

TABLE 4.20 – IvPs de l'effort des nouveaux projets

Projet	Taille fonctionnelle estimée	IvP cas(a)	IvP cas(b)	Taille fonctionnelle réelle	Effort réel
1	Triang(130,145,180)	[576,633]	[555,716]	159	677
2	Unif(300,400)	[542,613]	[522,647]	344	538
3	Unif(150,190)	[663,826]	[588,954]	179	845

4.3 Implémentation informatique

L'application web EstimSet implémente l'approche de sélection de modèle d'estimation pour une base de données telle qu'elle a été décrite dans le chapitre 2. Elle s'appuie sur un fichier de la base de données chargé par l'utilisateur et permet :

— de décrire les statistiques de la base de données entière.

— d'analyser et visualiser l'influence et la corrélation des attributs avec la variable dépendante (à estimer) telle que l'effort.

— de nettoyer et préparer la base de données.

— d'évaluer un modèle d'estimation ou évaluer et comparer un ensemble de modèles d'estimation parmi quatre modèles d'estimation proposés.

— de visualiser les performances d'un ou plusieurs modèles d'estimation évalués.

— de construire et sauvegarder un ou plusieurs système d'estimation, calibré(s) sur la base de données étudiée, à partir des modèles d'estimation évalués.

— d'utiliser le système d'estimation sauvegardé dans l'estimation de nouveaux projets.

Pour la programmation d'EstimSet, nous avons utilisé principalement les langages :

— Java (1.8) pour développer les interfaces et les requêtes.

— Langage "R" pour les analyses statistiques et traitements de modèles d'estimation

Le langage R est utilisé pour sa grande capacité de calcul et ses nombreux outils puissants d'analyse statistique.

Dans ce qui suit, nous détaillons l'architecture de cette application et nous déroulons un cas d'utilisation afin d'illustrer et valider les fonctionnalités programmées.

4.3.1 Architecture

L'application EstimSet est une application **REST** (Representational State Transfer), c'est à dire elle utilise le style d'architecture REST qui permet d'intégrer les fonctions de cette application dans des applications tierces ou des serveurs externes. Ainsi, l'application EstimSet peut être utilisée comme un module dans la plateforme d'estimation développée dans le cadre du projet Projestimate ou dans un service externe appelé par Web Service. La figure 4.10 présente l'architecture de l'application web EstimSet développée dans le cadre de cette thèse.

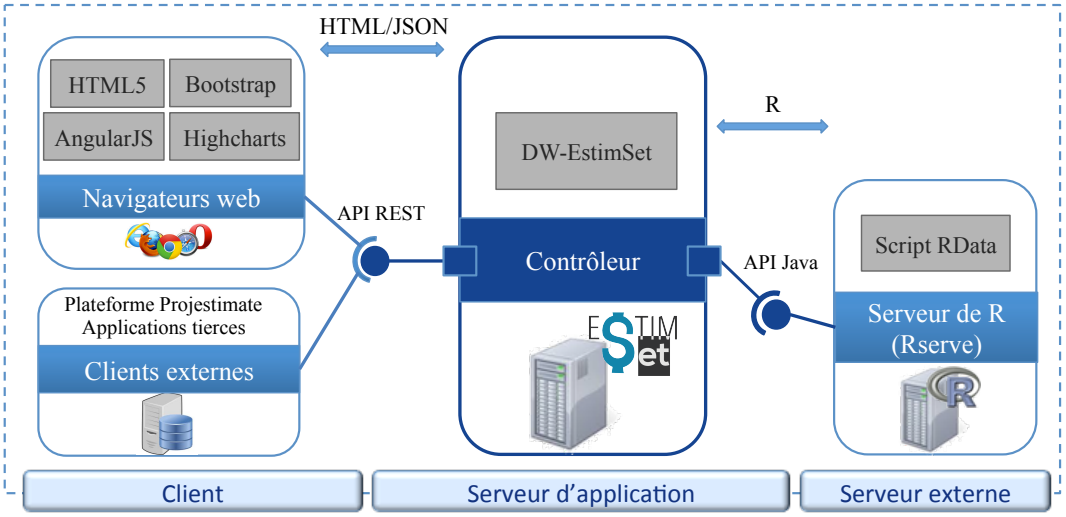


FIGURE 4.10 – Architecture de l'application Web EstimSet

L'architecture de l'application EstimSet se compose de trois composants : client, serveur d'application et serveur externe. Nous détaillons ci-après quelques outils utilisés du coté de chacun des composants.

Coté client : L'application est accessible via un navigateur web ou des clients externes. Différents outils sont utilisés coté client, à savoir HTML5, Bootstrap Twitter, AngularJS et Highcharts. L'HTML5 (HyperText Markup Language) est un langage de balisage permettant d'écrire de l'hypertexte et de représenter les page web. Bootstrap Twitter est un framework frontal (interfaces utilisateurs) intuitif et puissant pour un développement web rapide et facile. Il contient des codes HTML et CSS, formulaires, boutons, outils de navigation et autres éléments interactifs, ainsi que des extensions JavaScript en option. AngularJS est un framework JavaScript libre et open-source permettant de créer des applications web dynamiques. Elle permet, entre autre, de faire la communication entre le navigateur et le serveur. Highcharts est une bibliothèque de cartographie écrit en JavaScript. Elle offre un moyen facile d'ajouter des graphiques interactifs au sites web ou aux applications web. Highcharts permet de construire une grande variété de graphiques : ligne, spline, région, colonne, bars, etc.

Coté serveur d'application : Le serveur d'application se charge de la gestion et de la synchronisation des événements. Il utilise le framework Dropwizard et implémente quelques composants spécialisés tels que : Jetty, Jersey et Jackson.

Coté serveur externe : Le serveur Rserve permet aux logiciels ou applications d'utiliser les fonctions et méthodes de R à travers différents langages tel que Java. Chaque connexion a un espace de travail et un répertoire de travail distincts. Les implémentations côté client sont disponibles pour les langages populaires tels que C / C ++, PHP et Java. Rserve supporte la connexion à distance, l'authentification et le transfert de fichiers. L'utilisation typique est d'intégrer le langage R pour l'analyse des données et calculs statistiques dans d'autres applications. Il permet de transmettre des requêtes binaires à R [Urbanek2003].

Ainsi, le fonctionnement de l'application web EstimSet peut être simplement décrit selon 4 étapes :

— L'utilisateur accède à l'application web via un navigateur web et envoie sa requête (par exemple le chargement d'une base de données ou l'évaluation d'un modèle). Le navigateur envoie les requêtes en données de type JSON au contrôleur.

— Le contrôleur analyse les données reçues, s'il s'agit d'une action à effectuer dans R, il convertit les données en données valides de type R (par exemple RList, RBool) avant de les envoyer au serveur R.

— Au niveau du serveur R, le calcul se fait en utilisant des scripts de R contenant les fonctions d'analyse et de traitement nécessaires. Après avoir exécuté le traitement désiré , le serveur R renvoie la réponse de type R au contrôleur.

— La dernière étape consiste à convertir la réponse en données JSON par le contrôleur et l'envoyer ensuite au navigateur. L'utilisateur reçoit ainsi la réponse à sa requête.

4.3.2 Licence logicielle

Dans notre application, nous utilisons des librairies et serveurs tiers. Comme notre application se base principalement sur les fonctions de R, la majorité des librairies utilisées sont des librairies de R.

L'ensemble des composants de l'application est soumis aux licences GPL v2 (General Public License version 2) et GPL v3.

La licence GPL est destinée à garantir aux utilisateurs (développeur informatique, organisations, entreprises) la liberté d'utiliser, de partager et de modifier les logiciels ou codes sources sous conditions. Les logiciels et composants permettant ces droits sont qualifiés libres. La GPL permet de protéger les droits de l'utilisateur de deux façons : (1) faire valoir les droits d'auteurs sur le logiciel et (2) offrir à l'utilisateur cette Licence qui lui donne l'autorisation légale de copier, distribuer et/ou modifier le logiciel. Elle empêche donc la transformation d'un logiciel libre en logiciel propriétaire. C'est la licence de plus de la moitié de l'ensemble des logiciels libres actuellement distribués. La version 3 est la plus récente [FSF2015].

Étant donnée que la version 2 de GPL est la version la plus restrictive, notre application logicielle est libre et soumise à GPL v2.

4.3.3 Déroulement d'un cas d'utilisation

Afin d'illustrer les fonctionnalités développées dans l'application EstimSet, nous présentons le déroulement d'un cas d'utilisation étape par étape. L'interface contient deux onglets : un onglet "Compare/Generate model" pour la sélection de modèle d'estimation adéquat et la construction du système d'estimation, et un onglet "New Estimation" pour l'utilisation d'un système d'estimation pour l'estimation de l'effort de nouveaux projets (voir figure 4.11).

Nous nous focalisons, dans la suite de ce déroulement de cas d'utilisation, sur la procédure de sélection d'un modèle d'estimation et génération d'un système d'estimation réaliste.

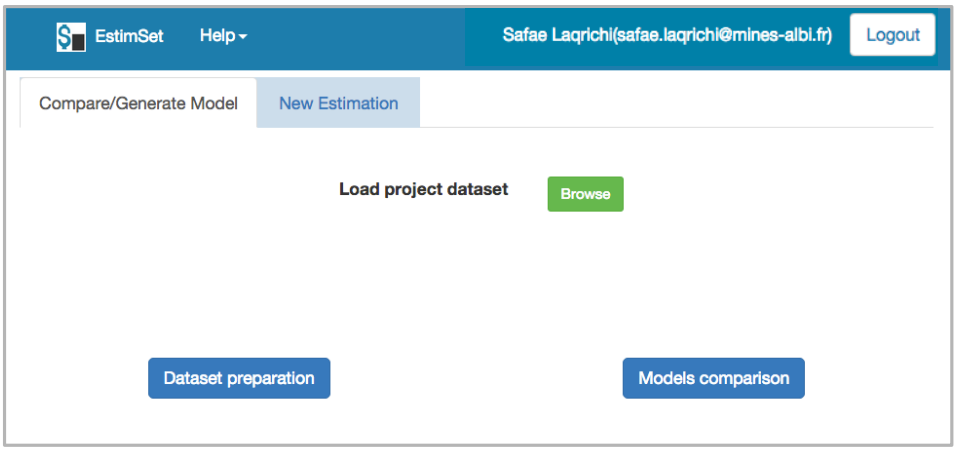


FIGURE 4.11 – Copie de l'écran de l'interface d'accueil de l'application EstimSet

L'onglet "Compare/Generate model" présente deux boutons : un bouton "Dataset preparation" permettant de préparer la base de données et un bouton "Models comparison" permettant de comparer des modèles d'estimation et de construire un système d'estimation.

La procédure pour comparer différents modèles d'estimation sur une base de données, commence tout d'abord par le chargement du fichier contenant la base de données (le fichier doit être sous le format ".csv"). Une fois chargé, l'application analyse les données et affiche les statistiques descriptives. Elle permet aussi de visualiser l'histogramme de chaque attribut (variable) (voir figure 4.12).

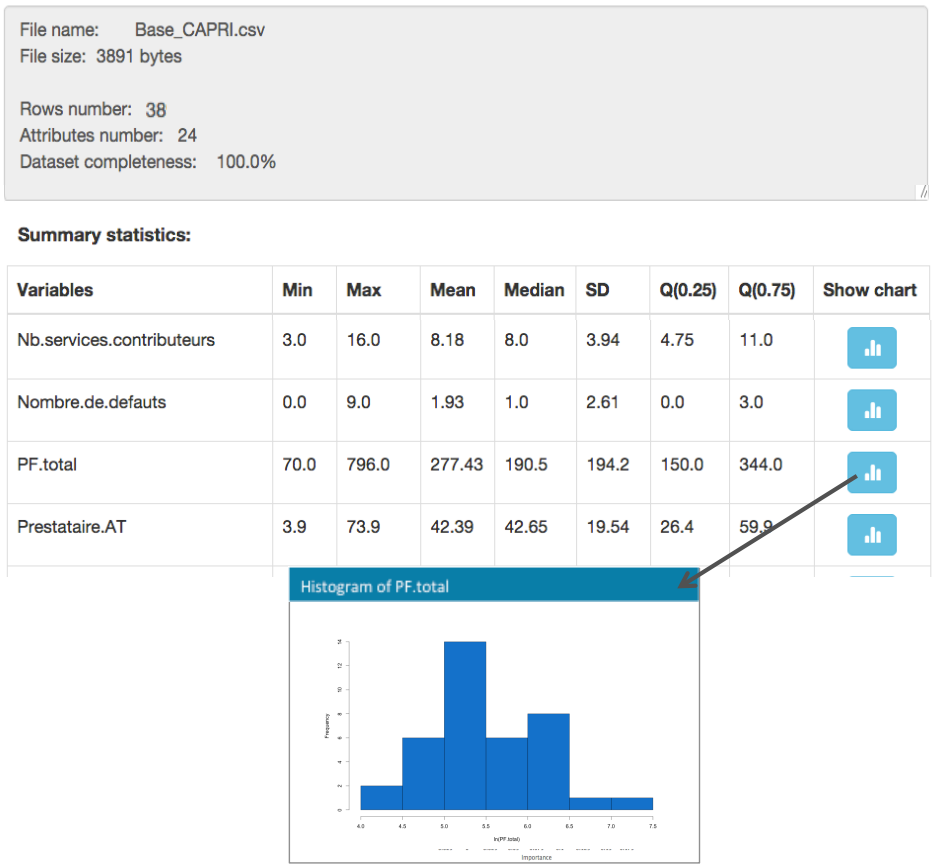


FIGURE 4.12 – Copie d'écran de l'étape de statistiques descriptives de la base de données dans l'application EstimSet

Le bouton "Dataset preparation" dirige ensuite l'utilisateur vers une nouvelle fenêtre (voir figure 4.13) permettant à l'utilisateur de :

1. Choisir les attributs qu'il estime déterminants de l'effort et la variable à estimer (par exemple l'effort).
2. Analyser les corrélations et l'importance de chaque attribut sélectionné ((voir figure 4.14).
3. Visualiser le graphique des importances des attributs sélectionnés afin de choisir les facteurs d'effort qui sont les plus influents (voir figure 4.14).

La sélection définitive des facteurs d'effort se fait par l'utilisateur en décochant les attributs que l'analyse a montré non importants. Une fois les facteurs d'effort sélectionnés, l'application se charge du nettoyage de la base de données des attributs non importants, des valeurs manquantes et des projets aberrants. Elle effectue également toutes les transformations nécessaires sur la base de données afin de permettre l'application de différents modèles d'estimation.

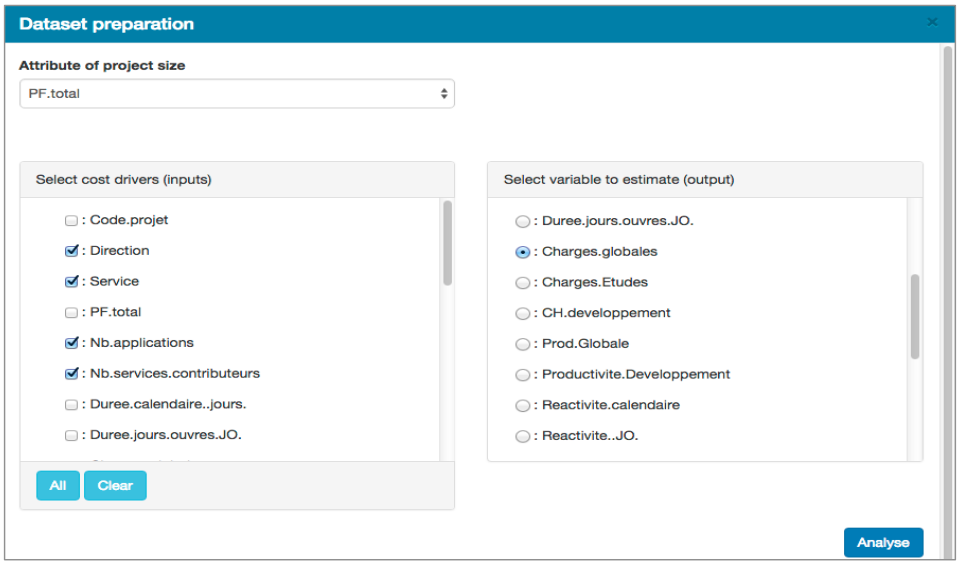


FIGURE 4.13 – Copie de l'écran de la fenêtre de sélection des facteurs d'effort de la base de données

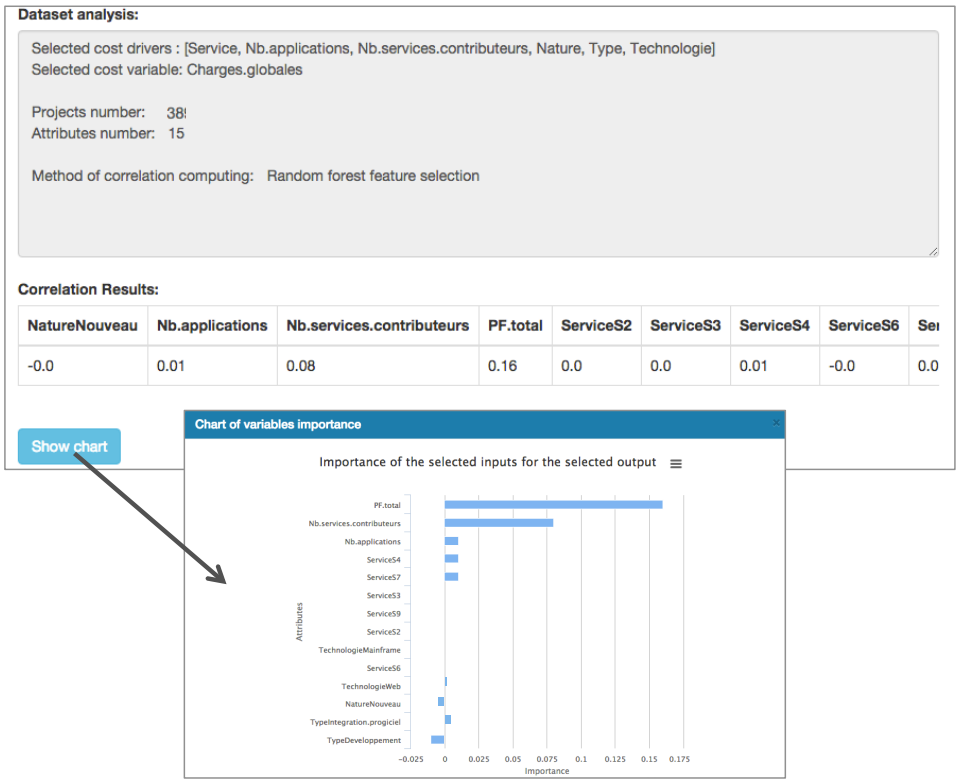


FIGURE 4.14 – Copie de l'écran des analyses de corrélation et d'importance des attributs sélectionnés

L'étape suivante est la comparaison des modèles d'estimation. Le bouton "Dataset preparation" dirige l'utilisateur vers une nouvelle fenêtre (voir figure 4.15) et l'invite à sélectionner les modèles d'estimation candidats parmi quatre modèles d'estimation et à déterminer les paramètres de

l'évaluation. Ces paramètres sont k_{int}, k_{ext} et B et correspondent aux procédures VCK, VCI et bootstrap.

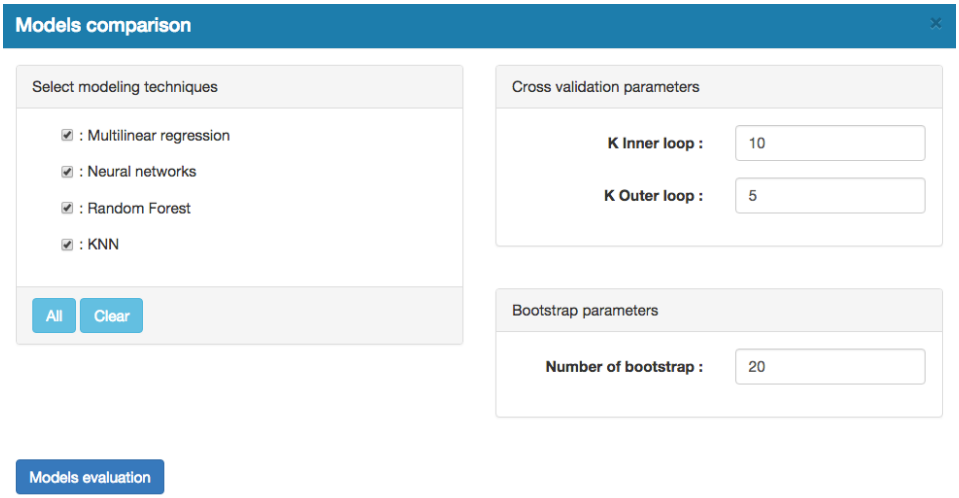


FIGURE 4.15 – Copie de l'écran de la fenêtre de sélection de modèles d'estimation

Le bouton "Models evaluation" permet de démarrer la procédure d'évaluation des modèles d'estimation sélectionnés selon l'approche que nous avons proposée (voir figure 4.16). Il affiche ensuite la table des résultats d'évaluation. EstimSet permet aussi de visualiser le graphe des valeurs estimées et valeurs réelles ainsi que le diagramme en boites à moustaches des *MRE* pour chaque modèle d'estimation sélectionné.

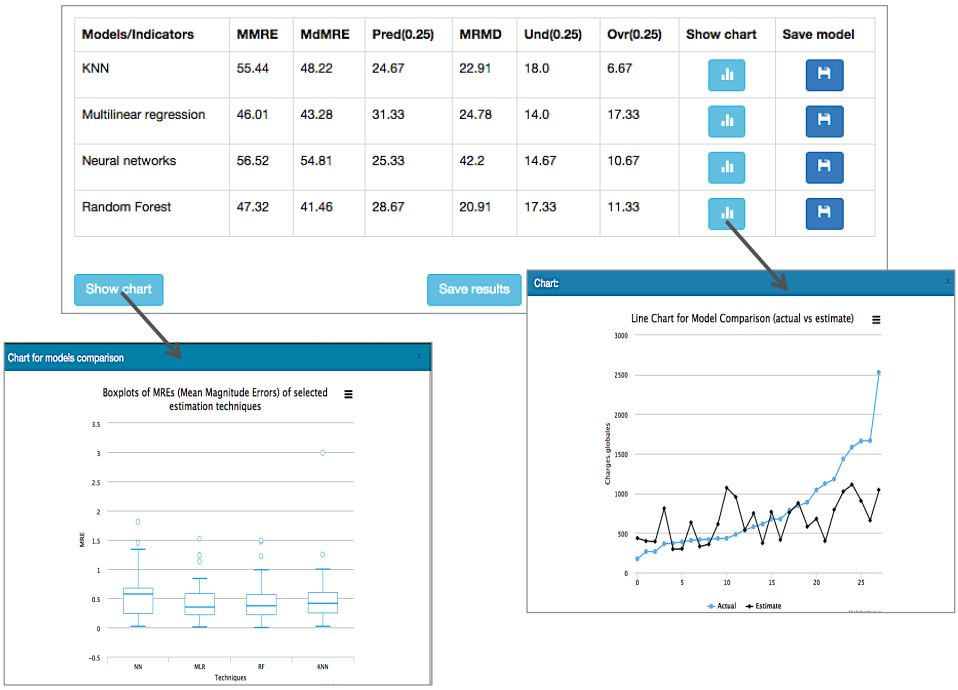


FIGURE 4.16 – Copie de l'écran des résultats de l'évaluation de modèles d'estimation sélectionnés

En s'appuyant sur ces résultats, l'utilisateur compare et sélectionne le modèle d'estimation le plus adéquat à son environnement et objectifs. La construction d'un système d'estimation réaliste à partir d'un modèle d'estimation se fait simultanément avec sa sauvegarde en cliquant le bouton "Save model" correspondant au modèle sélectionné.

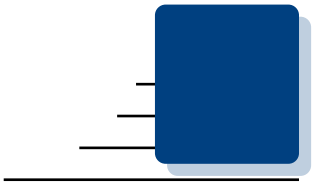
Conclusion

Nous avons expérimenté et validé l'approche de sélection de modèle d'estimation sur cinq bases de données différentes. Les résultats ont montré, d'une part, que les modèles d'estimation présentent différentes performances sur chacune des bases de données. D'autre part, ils ont montré que la sélection d'un modèle d'estimation n'est pas toujours une tâche évidente. En effet, certains modèles d'estimation montrent des performances comparables sur plusieurs bases de données. Dans ce cas, l'utilisateur doit prioriser les indicateurs en fonction de ses objectifs et besoins.

L'expérimentation montre également que le RLM et le FAD présentent souvent les meilleures performances. Le modèle d'estimation FAD n'est pas encore suffisamment exploré dans le domaine de l'estimation de l'effort de développement logiciel. Les résultats de cette expérimentation incitent à approfondir les recherches et études sur ce modèle d'estimation.

Nous avons développé dans le cadre de cette thèse, une application web EstimSet permettant d'automatiser le framework d'estimation que nous avons proposé et plus particulièrement l'approche de sélection du modèle d'estimation le plus adéquat pour une base de données. L'application est développée selon le style d'architecture REST, ce choix architectural permet une facilité d'intégration et d'utilisation des fonctionnalités de l'application dans d'autres applications Web. EstimSet est soumise à la licence GPL v2, due à l'utilisation des librairies soumises à la même licence et surtout les librairies de R.

EstimSet fournit l'ensemble des outils d'analyse statistique et de modélisation permettant de supporter l'utilisateur tout au long du processus de l'estimation de l'effort. Nous avons illustré ces outils sur un cas d'utilisation simple qui a mis en exergue l'intérêt d'une telle application.



Conclusion générale

Bilan

L'estimation réaliste de l'effort, des coûts et des délais de projets de développement logiciel est d'une importance capitale dans le succès des organisations de développement. C'est une des activités majeures de management de projet. Elle permet, entre autres, de répondre aux appels d'offre, de négocier les contrats, et de contrôler et suivre les projets. L'estimation de l'effort de développement logiciel est une activité complexe du fait de l'implication de différentes pratiques et éléments difficiles à gérer.

Depuis les années 1980, le développement informatique a connu des évolutions importantes en ce qui concerne les systèmes d'exploitation (création de l'UNIX, Windows, Android ···), les méthodes de conception (conception par objets, méthodes agiles ···), et les langages de programmation (Perl, Java, PHP ···), etc. Ces évolutions ont fait que les méthodes (formelles) génériques et classiques telles que la méthode COCOMO sont devenues inadaptées pour refléter l'état actuel et produire des estimations réalistes d'effort de nouveaux projets. Ces méthodes ont été construites à partir de bases de données anciennes ne couvrant pas les nouveaux types, contextes et environnements de projets actuels. Il est donc important de pouvoir construire de nouveaux modèles d'estimation permettant de prendre en compte ces nouveaux contextes. De plus, il est pertinent pour les entreprise de pouvoir construire des systèmes d'estimation propres à leurs contextes et besoins afin d'assurer des estimations d'effort les plus réalistes et pertinentes possible. La construction d'un système d'estimation s'appuie sur un modèle d'estimation basé sur les données (technique de modélisation), sélectionné parmi la multitude de modèles d'estimation existants.

Les projets de développement logiciel, en particulier pour ce qui concerne l'activité d'estimation, se caractérisent par la présence d'importantes incertitudes, surtout en phases amont. Nous avons identifié trois principales sources d'incertitudes qui sont relatives (1) au modèle d'estimation, (2) à la base de données utilisée pour calibrer le modèle et (3) aux nouveaux projets dont les informations sont imprécises et incomplètes et les spécifications sont changeantes au cours du projet. Ainsi, pour obtenir des estimations réalistes, il est important de pouvoir tenir compte de ces incertitudes tout au long du processus d'estimation.

C'est dans ce cadre que cette thèse propose une approche globale permettant de guider les organisations dans la construction d'un système d'estimation réaliste de l'effort (sélection et calibrage de modèle d'estimation) et de son utilisation, tout en tenant compte des incertitudes. Cette approche apporte des réponses aux questions Q.1, Q.2 et Q.3, soulevées dans l'introduction :

Q.1 Comment sélectionner, d'une manière fiable et en tenant compte des incertitudes, un modèle d'estimation pour le contexte particulier d'une organisation ?

Q.2 Comment construire un système d'estimation réaliste ?

Q.3 Comment utiliser le système construit pour les estimations de l'effort de nouveaux projets présentant des incertitudes ?

Nos contributions respectivement par rapport à ces questions sont détaillées ci-après.

Sélection d'un modèle d'estimation d'effort

Face aux nombreux modèles d'estimation existants basés sur les données, il est important pour l'entreprise de sélectionner le plus adéquat à son contexte. Les différents travaux de comparaison de modèles d'estimation existants dans la littérature prouvent que malheureusement il n'existe pas de modèle d'estimation unique performant pour tout contexte et environnement. De surcroit, quelques travaux montrent des résultats contradictoires lors de la comparaison d'un ensemble de modèles d'estimation sur la même base de données. Ainsi, il est parfois difficile de conclure sur le modèle d'estimation le plus adapté pour une base de données spécifique. Les raisons de ces contradictions sont relatives à la procédure de comparaison, y compris la méthode de validation et les critères de sélection utilisés. La technique de validation croisée k-fold la plus communément utilisée est prouvée pour être biaisée dans le cas des modèles d'estimation non paramétriques. Quant aux critères de sélection, d'une part, quelques uns sont biaisés et ne représentent pas bien la performance prédictive des modèles d'estimation. D'autre part, il n'existe pratiquement pas de critère qui quantifie les incertitudes des modèles d'estimation sur la base de données étudiée.

Nous avons proposé une approche globale permettant d'évaluer les modèles d'estimation d'une façon fiable et non biaisée en tenant compte des incertitudes. L'évaluation fiable est assurée en utilisant la technique de validation croisée imbriquée, en particulier pour les modèles d'estimation non paramétriques. Cette technique permet de réduire les biais liés à l'évaluation de ces modèles par la technique validation croisée k-folds. Nous avons proposé plusieurs indicateurs de performance comme critères de sélection. Ces indicateurs mesurent la performance prédictive ainsi que les incertitudes des modèles d'estimation sur la base de données étudiée. Ils permettent ainsi de donner une vision globale sur les différents aspects des modèles d'estimation évalués, y compris leurs incertitudes. La sélection d'un modèle d'estimation n'est pas toujours évidente, il s'agit d'un problème multi-dimensionnel dans la mesure où il se base sur plusieurs indicateurs de performance. Ainsi, pour guider l'utilisateur dans cette tâche nous avons proposé des orientations. L'utilisateur peut privilégier et favoriser, en fonction de ses besoins et objectifs, un indicateur de performance particulier afin de sélectionner le modèle d'estimation le plus adéquat. Ainsi, notre approche offre une grande flexibilité à l'utilisateur.

Nous avons par ailleurs expérimenté notre approche de sélection de modèle d'estimation, sur cinq bases de données différentes. Quatre modèles d'estimation ont été évalués et comparés : RLM, RNA, FAD, k-PPV. Les résultats ont montré que les performances de chaque modèle d'estimation varient d'une base de données à une autre. L'expérimentation montre également que le modèle simple basé sur la RLM ainsi que le modèle de type FAD présentent dans 80% des cas (40% des cas chacun) les meilleures performances.

Construction d'un système d'estimation performant à partir du modèle d'estimation sélectionné

Pour construire un système d'estimation propre à l'organisation, l'approche traditionnelle consiste à calibrer le modèle sélectionné sur la base de données de l'organisation. Les estimations d'effort obtenues sont ponctuelles et ne permettent pas d'exprimer les incertitudes inhérentes au système d'estimation (incertitudes relatives à la base de données et au modèle d'estimation sélectionné). Différentes techniques ont été proposées pour exprimer les incertitudes du système d'estimation dans les estimations en termes d'intervalles de prédiction, notamment l'approche de JORGENSEN et al. [Jorgensen+2003] basée sur l'expérience de l'utilisateur, et le bootstrap.

Nous avons définit un système d'estimation performant comme étant un système d'estimation permettant de (1) représenter les incertitudes inhérentes aux éléments intervenant dans sa construction (base de données, modèles d'estimation) et (2) produire des estimations d'effort les plus proches possibles des efforts réels.

Contrairement à l'approche traditionnelle qui s'appuie sur un seul modèle d'estimation de l'effort calibré, nous avons proposé de construire un système d'estimation composé de plusieurs instances (aussi appelées "répliques") du modèle d'estimation sélectionné. Pour construire ces instances, nous avons choisi d'utiliser la technique de ré-échantillonnage "bootstrap" non paramétrique. Cette technique présente l'avantage de ne pas nécessiter une expérience dans l'estimation et d'être facilement automatisable. Le système d'estimation résultant est multi-modèles, il permet d'intégrer et représenter les incertitudes ainsi que de produire des estimations réalistes. Notre approche permet également de mettre à jour le système d'estimation pour s'adapter à l'évolution du contexte. Pour cela, elle doit être appliquée sur la base de données actualisée afin d'intégrer les nouvelles informations dans la phase de sélection de modèle d'estimation ainsi que dans la construction du nouveau système d'estimation.

Utilisation du système d'estimation construit pour l'estimation d'un nouveau projet

Les nouveaux projets de développement logiciel présentent d'importantes incertitudes, surtout en phases amont. Nous nous sommes intéressé alors à prendre en compte ces incertitudes dans l'estimation de l'effort. Ce type d'incertitude est peu abordé dans le domaine de l'estimation de l'effort de développement. Il est surtout abordé et quantifié dans le domaine de la métrologie où des techniques telles que l'analyse de sensibilité et Monte Carlo sont utilisées.

En raison l'adéquation de la méthode Monte Carlo au problème traité, nous l'avons utilisée dans notre approche pour l'estimation de nouveaux projets afin de prendre en compte les incertitudes qui les caractérisent. Dans notre approche, les incertitudes relatives aux facteurs d'effort décrivant le nouveau projet à estimer sont exprimées à travers des fonctions de densité de probabilité. Ces fonctions sont ensuite propagées dans l'effort estimé à travers le système d'estimation en utilisant Monte Carlo. Notre contribution à ce niveau porte sur la manière de propager les incertitudes des facteurs d'effort dans un système d'estimation multi-modèles et de dériver l'intervalle de prédiction intégrant les trois types d'incertitudes.

Une des limitations de notre approche est relative à l'évaluation subjective des FDPs des facteurs d'effort par l'utilisateur. Cette subjectivité est d'autant plus importante en phases amont du projet et influence, par conséquent, fortement la précision de la distribution obtenue de l'effort estimé.

L'ensemble des techniques et procédures statistiques présentées dans ces travaux de thèse ont été implémentées dans une application Web basée sur les langages Java et R. Cet outil permet d'aider l'utilisateur dans la construction d'un modèle d'estimation adapté à son contexte et à ses objectifs, ainsi que dans son utilisation pour estimer l'effort de nouveaux projets. Nous pouvons facilement imaginer d'autres usages à cet outil, notamment pour l'estimation dans d'autres domaines d'industrie tels que la construction et la production.

Perspectives de recherche

Nous présentons dans cette section quelques pistes principales de recherche ainsi que des propositions permettant d'améliorer les travaux développés dans cette thèse ou bien d'étendre leurs utilisations à d'autres contextes.

Enrichissement du système d'estimation

Pour améliorer la performance du système d'estimation construit, il serait intéressant d'étudier l'influence de la combinaison des modèles d'estimation les mieux classés sur la performance de l'estimation. Quelques travaux mentionnés dans la littérature, principalement [Mittas+2010 ; Mittas+2015b] ont été menés dans ce sens en explorant le couplage de modèles d'estimation non-paramétriques tels que RNA avec des modèles d'estimation paramétriques tel que RLM.

Ces recherches ont amené à introduire un nouveau type de modèles d'estimation appelé semi-paramétriques et leurs résultats ont montré que ces nouveaux modèles ont des performances meilleures comparés aux autres modèles d'estimation. Différents couplages ont été testés sur les bases ISBSG et Nasa93 mais aucun n'utilise le modèle d'estimation non paramétrique FAD. Ce modèle, qui n'est pas encore suffisamment exploré dans le domaine de l'estimation de l'effort de développement logiciel, a montré de bonnes performances dans nos expérimentations. Il serait donc intéressant d'enrichir notre approche en explorant, pour chaque base de données étudiée, les différents couplages possibles et éventuellement se focaliser sur l'étude du comportement et de la performance du couplage des deux modèles d'estimation de type FAD et de type RLM.

Dans la même optique d'enrichissement, une autre perspective serait de faire évoluer le système d'estimation afin de permettre l'actualisation et la révision des estimations tout au long du cycle de vie de projet afin de mieux le contrôler. Un exemple de méthode d'estimation offrant cette opportunité est COCOMO II [Boehm2000] qui consiste en deux modèles d'estimation : un pour la phase début de conception (architecture) et l'autre pour la phase post-conception. En avançant dans le projet, le système d'estimation à utiliser est de plus en plus détaillé en raison de la disponibilité de plus d'information par rapport aux phases ultérieures. En s'inspirant de ce modèle, nous proposons d'étudier la possibilité et la pertinence de construire différents systèmes d'estimation destinés chacun à une phase définie du cycle de vie du projet logiciel. Pour cela, nous suggérons de préparer différentes bases de données, chacune dédiée à la construction d'un système d'estimation pour une phase définie de projet et qui ne contient comme facteurs d'effort que ceux qui peuvent être estimés dans cette phase. Ainsi, il est important que la base de données brute de l'organisation contienne des facteurs d'effort généraux (facilement connus en phases amonts de projet) et d'autres détaillés.

Prise en compte des risques dans l'estimation de nouveaux projets

Les risques sont définis comme des évènements susceptibles de perturber le projet et par voie de conséquence l'effort qu'il nécessite pour son développement ainsi que son coût. Ils sont naturellement présents aussi bien que les incertitudes, dans le processus de l'estimation de l'effort. Il est donc important de les prendre en compte dans l'estimation de l'effort.

Nous proposons, pour ce propos, de nous appuyer sur les recherches effectuées par notre équipe sur la gestion des risques des projets de développement de nouveaux produits dans le cadre du projet ProRisk [Marmier+2013 ; Nguyen+2013]. Ce projet offre aux chefs de projet un outil d'aide à la décision, permettant : (1) d'évaluer globalement le niveau de risque d'un projet, c'est à dire la chance que le projet satisfasse les engagements et (2) d'aider à la sélection des meilleures stratégies de traitement de risques. L'approche Prorisk se compose de trois phases principales :

— Modélisation du projet, risques et stratégies de traitement : consiste à planifier le projet, identifier et évaluer les risques potentiels et les stratégies de traitement.

— Génération des scénarios possibles et évaluation de chaque scénario en termes de différentes critères tels que la probabilité, le coût et le délai.

— Prise de décision : en s'appuyant sur la visualisation des différents scénarios sous la forme de graphes, l'utilisateur peut conduire son analyse sur le niveau de risque de son projet ainsi que sur les stratégies de traitement de risques à mettre en place.

Il serait donc envisageable d'adapter et utiliser l'approche ProRisk dans le cas des projets de développement logiciel. Nous suggérons ainsi la procédure suivante :

— Estimer l'effort de projet : selon le système d'estimation construit, il peut s'agir de l'effort global ou d'un effort d'un phase ou d'une activité du projet

— Dériver l'effort de chacune des tâches ou phases de projet : des abaques de répartition de l'effort peuvent être utilisées pour cela.

— Construire le planning du projet

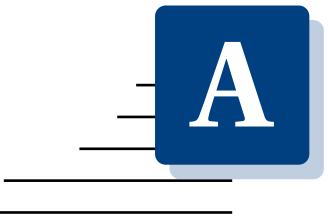
— Identifier et évaluer les risques et les stratégies de traitement possibles du projet : pour l'identification des risques nous pouvons nous appuyer sur des typologies de risques telles que la liste de WALLACE et al. [Laqrichi+2013b ; Wallace+2004].

— Génération et évaluation (probabilité, coût et délai) des scénarios possibles

— Analyser le coût et le délai des différents scénarios, sélectionner des stratégies de traitement adéquates et dériver des estimations réalistes de coût et de délai prenant en compte les risques.

Annexes

ANNEXE



Modèle COCOMO

Les modèles COCOMO 81 reposent sur l'équation :

$$\widetilde{E} = a(KLOC)^b \times FAE \tag{A.0.1}$$

où $\widetilde{E}$ est l'effort estimé en mois-homme, KLOC est la taille du projet logiciel estimée en kilo (mille) lignes de code source. Les paramètres a et b sont des constantes qui dépendent de la complexité du logiciel (organique, semi-détaché ou intégré) et peuvent être déterminés à partir de la table A.1. Les types/complexité de projet sont :

— Organique : Application simple, de routine, réalisée par une équipe expérimentée ayant l'habitude de travailler ensemble, maîtrisant le langage et l'environnement de développement

— Semi-Détaché : Niveau intermédiaire, le projet n'est ni trop simple ni trop compliqué, l'équipe de développement a déjà réalisé quelques projets ensemble mais n'est pas totalement rodée. Les technologies et le domaine d'application sont un peu flous, mais pas de grosses difficultés.

— Imbriqué : Techniques innovantes, organisation complexe, couplage fort avec beaucoup d'interactions. Technologie et domaine nouveau, équipe 'jeune'.

FAE est le facteur d'ajustement de l'effort, il est ajusté selon le modèle COCOMO 81 utilisé comme suit :

$$FAE = \begin{cases} 1 & \textit{(modèle basique)} \\ \prod_{i=1}^{15} C_i & \textit{(modèle intermédiaire et détaillé)} \end{cases} \tag{A.0.2}$$

Où C_i représente la valeur du facteur d'effort i qui peut être déterminée à partir de la table A.2.

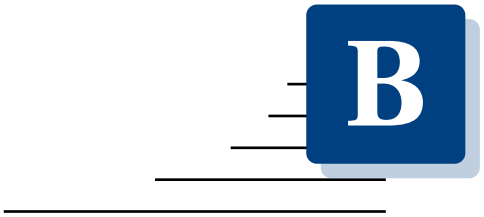
TABLE A.1 – Paramètres a et b en fonction du type/complexité du projet

Type/complexité de projet	a	b
Organique	2.4	1.05
Semi-détaché	3.0	1.12
Intégré	3.6	1.20

TABLE A.2 – Facteurs d'effort C_i des modèles COCOMO intermédiaire et détaillé

Ratings/Niveaux						
Facteurs de Cots Cost Drivers	Very Low	Low	Nominal	High	Very High	Extra High
Product Attributes (Complexité de l'application et/ou du produit à réaliser)						
RELY Required software reliability	0.75	0.88	1	1.15	1.4	
DATA Data base size		0.94	1	1.08	1.16	
CPLX Product complexity	0.7	0.85	1	1.15	1.3	1.65
Computer Attributes (Qualite de service du centre de calcul et de l'ordinateur cible)						
TIME Execution time constraint			1	1.11	1.3	1.66
STOR Main storage constraint			1	1.06	1.21	1.56
VIRT Virtual machine volatility		0.87	1	1.15	1.3	
TURN Computer turnaround time		0.87	1	1.07	1.15	
Personnel Attributes (Expérience et/ou maturité des programmeurs individuellement et collectivement)						
ACAP Analyst capability	1.46	1.19	1	0.86	0.71	
AEXP Applications experience	1.29	1.13	1	0.91	0.82	
PCAP Programmer capability	1.42	1.17	1	0.86	0.7	
VEXP Virtual machine experience	1.21	1.1	1	0.9		
LEXP Programming language experience	1.14	1.07	1	0.95		
Project Attributes (Processus et ou méthodes de développement adoptés par le projet)						
MODP Use of modern programming practices	1.24	1.1	1	0.91	0.82	
TOOL Use of software tools						
SCED Required development schedule	1.24	1.1	1	0.91	0.83	
	1.23	1.08	1	1.04	1.1	

ANNEXE



Bases de données de PROMISE

B.1 Description des attributs de la base de données Nasa93

La base de données Nasa93, aussi appelée COCOMO NASA 2, contient 93 projets développés entre 1971 et 1987 dans différents centres de la Nasa (National Aeronautics and Space Administration) aux États Unis. Cette base de données est très utilisée par les travaux de recherche dans le domaine de l'estimation de l'effort de projets de développement logiciel.

TABLE B.1 – Description des attributs de la base de données Nasa93

Attribut	Description	détails	Type
kloc	Kilo line of code	Taille du projet mesurée en kilo (1000) lignes de code source.	Numérique
rely	Reliability	Niveau de fiabilité requis.	Qualitatif
data	Data base size	Facteur relatif à la taille de la base de données, dérivé à partir du ratio D/P Ko/kloc).	Qualitatif
cplx	Complexity	mesure de la complexité du logiciel.	Qualitatif
time	Execution time constraints	Contraintes de temps d'exécution.	Qualitatif
stor	Main storage constraint	Contraintes de taille mémoire principale.	Qualitatif
virt	machine volatility	Instabilité (volatilité) de la machine virtuelle (système d'exploitation).	Qualitatif
turn	turnaround time	Temps de restitution des travaux.	Qualitatif
acap	Analyst capability	Compétence le l'équipe d'analyse.	Qualitatif
aexp	Analyst experience	Expérience dans l'application.	Qualitatif
pcap	Programmer capability	Compétence de l'équipe de programmation.	Qualitatif
vexp	Virtual machine experience	Expérience relativement à la machine virtuelle (système d'exploitation).	Qualitatif
lexp	Language experience	Expérience relative au langage de programmation.	Qualitatif
modp	modern programing practices	Pratique des méthodes modernes de programmation.	Qualitatif
tool	Use of software tools	Utilisation d'outils logiciels de développement et de gestion.	Qualitatif
sced	Contraintes de planning	Extension ou accélération de l'échéancier en pourcentage.	Qualitatif

B.2 Description des attributs de la base de données Desharnais

La base de données Desharnais contient 81 projets commerciaux développés au sein d'un éditeur de logiciels canadien en 1989.

TABLE B.2 – Description des attributs de la base de données Desharnais

Attribut	Description	détails	Type
TeamExp	Expérience de l'équipe	Mesuré en années	Numérique
ManagerExp	Expérience du manager	Mesuré en années	Numérique
YearEnd	Année de fin de projet	-	Qualitatif
Length	Échéancier du projet	Mesuré en mois (variable dépendante)	Numérique
Transactions	Nombre de transactions logiques dans le système	Mesuré en PF	Numérique
Entities	Nombre des entités dans le système	Mesuré en PF	Numérique
Envergure	Échelle du projet	Facteur d'ajustement de complexité de PF calculé à partir de 14 caractéristiques générales du système (GSC)	Numérique
PointsAdjust	Nombre des PF non ajustés	Mesuré en PF	Numérique
PointsNonAjust	Nombre des PF ajustés	Mesuré en PF	Numérique
Language	Langage de programmation	3 niveaux : 1, 2 et 3	Qualitatif

B.3 Description des attributs de la base de données Maxwell

La base de données Maxwell contient 62 projets développés par une des plus grande banque commerciale en Finlande de 1986 à 1993.

TABLE B.3 – Description des attributs de la base de données Maxwell

Attribut	Description	Type
App	Type d'application	Qualitatif
Har	Plateforme Hardware	Qualitatif
Dba	Type de base de données	Qualitatif
Ifc	Type de l'interface utilisateur	Qualitatif
Source	Lieu de développement	Qualitatif
Telonuse	Utilisation du Telon (AGL)	Qualitatif
Nlan	Nombre de langages	Numérique
T01	Participation du client	Numérique (ordinale)
T02	Convenance de l'environnement de développement	Numérique (ordinale)
T03	Disponibilité du personnel	Numérique (ordinale)
T04	Utilisation des standards	Numérique (ordinale)
T05	Utilisations des méthodes	Numérique (ordinale)
T06	Utilisation des outils	Numérique (ordinale)
T07	Complexité logique du logiciel	Numérique (ordinale)
T08	Volatilité des exigences	Numérique (ordinale)
T09	Exigences de qualité	Numérique (ordinale)
T10	Exigences d'efficience	Numérique (ordinale)
T11	Exigences d'installation	Numérique (ordinale)
T12	Compétence analytique du personnel	Numérique (ordinale)
T13	Connaissance du personnel sur l'application	Numérique (ordinale)
T14	Compétence du personnel dans les outils	Numérique (ordinale)
T15	Compétence de l'équipe	Numérique (ordinale)
Duration	Durée	Numérique
Size	Taille de l'application	Numérique
Effort	Effort	Numérique

Acronymes

AR AR 32
ARPI ARPI 29, 53

COCOMO Constructive Cost Model 16–18, 22, 23, 25, 28

EbA Estimation basé sur l'Analogie 34

FAD Forêts d'Arbres Décisionnels 23, 45
FDP Fonction de Densité de Probabilité 60–64, 66

GUM Guide to the Expression of Uncertainty in Measurement 23

HitRate HitRate 29, 53

IvP Intervalle de Prédiction 26–30, 57, 65, 67

k-PPV k-Plus Proches Voisins xi, 21, 23

MdMRE Median Magnitude of Relative Error 31, 32, 48–50, 55, 76, 77, 79, 81, 83
MMRE Mean Magnitude of Relative Error 31, 32, 34, 50, 55, 76, 77, 79, 81–83
MRE Magnitude of Relative Error xi, 31, 32, 34, 48, 50, 52, 55, 82, 93
MRMD Mean Relative Mean Difference xi, 54, 55, 76, 77, 79, 81–83

Ovr Overestimation proneness at level l 50, 55, 77, 81

PCU Points de Cas d'Utilisation 13
PF Points de Fonction 13, 22, 25
Pred Prediction accuracy at level l 31, 32, 50, 55, 76, 77, 79, 81, 83

RLM Régression Linéaire Multiple 18, 23, 34, 43, 44, 77, 97
RLS Régression Linéaire Simple 23
RMD RMD 53, 54

RMSE RMSE 32

RNA Réseaux de Neurones Artificiels 23, 34, 43, 44, 97

RWidth RWidth 29, 53

SLIM Software Lifecycle Management 16, 22, 23

SLOC Lignes de Code Source 22

Und Underestimation proneness at level l 50, 55, 77, 81

VCK Validation Croisée "k-fold" 33, 35, 59

Glossaire

biais C'est l'erreur systématique, il représente la distance entre une la valeur réelle et la valeur estimée. Dans le contexte de l'estimation de l'effort, le biais représente la distance entre l'effort réel et l'effort estimé. Le biais d'une méthode de validation est la différence entre la performance réel et la performance estimée/évaluée du modèle d'estimation par cette méthode de validation. 38

estimation réaliste Une estimation proche de la valeur réelle et qui reflète et représente les incertitudes présentes dans le monde réel. 57

hyper-paramètres Les hyper-paramètres d'un modèle non paramétrique sont les variables qui contrôlent la configuration de ce modèle. 44–48, 52, 58, 59

inconstance Disposition à se changer d'une étude à une autre. 34

non paramétriques Les modèles d'estimation non paramétriques sont les modèle d'estimation ayant une forme non déterminée à priori et un nombre de paramètres non fixé à priori. Les modèles non paramétriques ont, en plus des coefficients, des hyper-paramètres qui sont des variables contrôlant la configuration des modèles. 44–46, 59

paramétriques Les modèles d'estimation paramétriques sont les modèles ayant une forme bien définie et un nombre fixe de paramètres. Les paramètres des modèles d'estimation paramétriques sont appelés les coefficients. 44, 48, 58

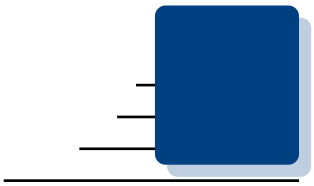
performance prédictive La capacité d'un modèle d'estimation à produire des estimations d'efforts proches des efforts réels.. 40, 44, 50, 51, 53, 55, 57, 59

performant Un modèle d'estimation performant est un modèle ayant une bonne capacité prédictive et qui présente peu d'incertitudes. Un système d'estimation performant est un système qui produit des estimations d'efforts proches des effort réels et représentatives de la réalité (sous forme probabiliste). 55, 56

réplique Modèle calibré sur un échantillon de bootstrap. 54

sous-modèle Modèle calibré sur une partition de la base de données (validation croisée "k-fold"). 46–49

système d'estimation Un modèle ou ensemble de modèles d'estimation calibrés sur la base de donnée étudiée. Il est prêt à être utiliser pour l'estimation d'efforts de nouveaux projets. 38, 57



Bibliographie

[Adams2002] Thomas M. ADAMS. « G104-A2LA Guide for estimation of measurement uncertainty in testing ». In : *American Association of Laboratory Accreditation Manual* (2002), p. 10–18 (cf. p. 25).

[Afifi+1979] A. A. AFIFI et S. P. AZEN. *Statistical Analysis, Second Edition: A Computer Oriented Approach*. English. 2 edition. New York : Academic Press, mar. 1979 (cf. p. 41).

[Albrecht1979] Aj ALBRECHT. « Measuring Application Development Productivity ». In : *Proc. of IBM Application Development Symp.* Sous la dir. d'IBM PRESS. Oct. 1979, p. 83–92 (cf. p. 13).

[Anda+2001] B. ANDA, H. DREIEM, D. SJØBERG et M. SJØBERG. « Estimating software development effort based on use cases-experiences from industry ». In : *«UML» 2001—The Unified Modeling Language. Modeling Languages, Concepts, and Tools* (2001), p. 487–502 (cf. p. 13).

[Andreou+2008] A.S. ANDREOU et E. PAPATHEOCHAROUS. « Software Cost Estimation using Fuzzy Decision Trees ». In : *23rd IEEE/ACM International Conference on Automated Software Engineering, 2008. ASE 2008.* Sept. 2008, p. 371–374 (cf. p. 21).

[Angelis+2000] L. ANGELIS et I. STAMELOS. « A simulation tool for efficient analogy based cost estimation ». In : *Empirical software engineering* 5.1 (2000), p. 35–68 (cf. p. 26–28).

[Arlot+2010] Sylvain ARLOT et Alain CELISSE. « A survey of cross-validation procedures for model selection ». en. In : *Statistics Surveys* 4 (2010), p. 40–79 (cf. p. 33).

[Armstrong2002] J. S. ARMSTRONG, éd. *Principles of Forecasting: A Handbook for Researchers and Practitioners*. English. 2001 edition. New York : Springer, jan. 2002 (cf. p. 27).

[Asadullah2015] Munshi ASADULLAH. « Identification of Function Points in Software Specifications Using Natural Language Processing ». Thèse de doct. France : Université de Paris-Sud, 2015 (cf. p. 12).

[Azzeh2012] Mohammad AZZEH. « A replicated assessment and comparison of adaptation techniques for analogy-based effort estimation ». en. In : *Empirical Software Engineering* 17.1-2 (fév. 2012), p. 90–127 (cf. p. 34).

[Bakır+2010] A. BAKIR, B. TURHAN et A. BENER. « A comparative study for estimating software development effort intervals ». In : *Software Quality Journal* 19.3 (sept. 2010), p. 537–552 (cf. p. 27).

[Baumann+2014] Désirée BAUMANN et Knut BAUMANN. « Reliable estimation of prediction errors for QSAR models under model uncertainty using double cross-validation ». In : *Journal of cheminformatics* 6.1 (2014), p. 47 (cf. p. 47).

[Boehm1981] Barry W. BOEHM. *Software engineering economics*. Prentice-Hall advances in computing science and technology series. Englewood Cliffs, N.J : Prentice-Hall, 1981 (cf. p. 15, 17, 26).

[Boehm2000] B. BOEHM. « Software Cost Estimation with Cocomo II ». In : *Prentice-Hall* (2000) (cf. p. 18, 98).

[Box+1964] G. E. P. BOX et D. R. COX. « An analysis of transformations ». In : *Journal of the Royal Statistical Society. Series B (Methodological* (1964), p. 211–252 (cf. p. 43).

[Breiman+1984] Leo BREIMAN, Jerome FRIEDMAN, Charles J. STONE et R. A. OLSHEN. *Classification and Regression Trees*. English. 1 edition. Boca Raton : Chapman et Hall/CRC, jan. 1984 (cf. p. 20).

[Breiman2001] Leo BREIMAN. « Random Forests ». en. In : *Machine Learning* 45.1 (oct. 2001), p. 5–32 (cf. p. 20).

[Briand+2002] L. C. BRIAND et I. WIECZOREK. « Resource estimation in software engineering ». In : *Encyclopedia of Software engineering* (2002) (cf. p. 15).

[Brooks2001] Frederick P. BROOKS. *Le mythe du mois-homme*. Français. 2e éd. Paris; Albany; Belmont : International Thomson publishing France, août 2001 (cf. p. 13).

[Brunet2010] Florent BRUNET. « Contributions to Parametric Image Registration and 3D Surface Reconstruction ». Thèse de doct. Université d'Auvergne, 2010 (cf. p. 17, 18).

[Cahan+2011] Sorel CAHAN et Eyal GAMLIEL. « First among others? Cohen'sd vs. alternative standardized mean group difference measures ». In : *Practical Assessment, Research and Evaluation* 16.10 (2011), p. 1–6 (cf. p. 53).

[CampionAwwad+2014] Oliver CAMPION-AWWAD, Alexander HAYTON, Leila SMITH et Mark VUARAN. « The National Programme for IT in the NHS ». In : (2014) (cf. p. 2).

[Capers2006] Jones CAPERS. *the economics of software maintenance in the twenty first century*. 2006 (cf. p. 4).

[Chandrashekar+2014] Girish CHANDRASHEKAR et Ferat SAHIN. « A survey on feature selection methods ». In : *Computers & Electrical Engineering*. 40th-year commemorative issue 40.1 (jan. 2014), p. 16–28 (cf. p. 42).

[Chemuturi+2010] Murali CHEMUTURI et Thomas M. CAGLEY. *Mastering Software Project Management: Best Practices, Tools and Techniques*. en. J. Ross Publishing, juil. 2010 (cf. p. 5, 14).

[Chemuturi2009] Murali CHEMUTURI. *Software Estimation Best Practices, Tools & Techniques: A Complete Guide for Software Project Estimators*. en. J. Ross Publishing, juil. 2009 (cf. p. 6).

[Chemuturi2013] Murali CHEMUTURI. *Requirements Engineering and Management for Software Development Projects*. en. New York, NY : Springer New York, 2013 (cf. p. 4).

[Chiu+2007] Nan-Hsing CHIU et Sun-Jen HUANG. « The adjusted analogy-based software effort estimation based on similarity distances ». In : *Journal of Systems and Software* 80.4 (avr. 2007), p. 628–640 (cf. p. 15).

[Cichosz2015] Pawel CICHOSZ. *Data Mining Algorithms: Explained Using R*. en. John Wiley & Sons, jan. 2015 (cf. p. 43).

[Clayton2014] Richard CLAYTON. *Software Estimation is a Losing Game*. 2014 (cf. p. 5).

[Conte1986] Samuel Daniel CONTE. *Software Engineering Metrics and Models*. Benjamin-Cummings Pub Co, mar. 1986 (cf. p. 31).

[Dave+2012] Vachik S. DAVE et K. DUTTA. « Neural network based models for software effort estimation: a review ». In : *Artificial Intelligence Review* 42 (mai 2012), p. 295–307 (cf. p. 20).

[Dienemann1966] Paul F. DIENEMANN. *Estimating cost uncertainty using Monte Carlo techniques*. DTIC Document, 1966 (cf. p. 63).

[Dreyfus+2011] Gérard DREYFUS, Jean-Marc MARTINEZ, Manuel SAMUELIDES, Mirta B. GORDON, Fouad BADRAN et Sylvie THIRIA. *Apprentissage statistique: Réseaux de neurones - Cartes topologiques - Machines à vecteurs supports*. fr. Editions Eyrolles, juil. 2011 (cf. p. 19).

[Efron+1994] Bradley EFRON et R. J. TIBSHIRANI. *An Introduction to the Bootstrap*. en. CRC Press, mai 1994 (cf. p. 28, 59).

[El Emam+2008] K. El Emam et A.G. Koru. « A Replicated Survey of IT Software Project Failures ». In : *IEEE Software* 25.5 (sept. 2008), p. 84–90 (cf. p. 4).

[Elyassami+2013] Sanaa Elyassami et A. Idri. « Evaluating software cost estimation models using fuzzy decision trees ». In : *Recent Advances in Knowledge Engineering and Systems Science, WSEAS Press* (2013), p. 243–248 (cf. p. 21).

[Everitt2010] Brian Everitt. *The Cambridge dictionary of statistics* /. 4th ed. Cambridge, UK ; Cambridge University Press, 2010 (cf. p. 41).

[Fairley1992] R.E. Fairley. « Recent advances in software estimation techniques ». In : *International Conference on Software Engineering, 1992.* 1992, p. 382–391 (cf. p. 17, 22).

[Fedotova+2013] Olga Fedotova, Leonor Teixeira et Helena Alvelos. « Software Effort Estimation with Multiple Linear Regression: Review and Practical Application ». In : *Journal of Information Science and Engineering* 29.5 (2013), p. 925–945 (cf. p. 18).

[Fiore+1996] Piernicola Fiore, Filippo Lanubile et Giuseppe Visaggio. « Effort estimation for program comprehension ». In : *Program Comprehension, 1996, Proceedings., Fourth Workshop on.* IEEE, 1996, p. 78–87 (cf. p. 18).

[Foss+2003] T. Foss, E. Stensrud, B. Kitchenham et I. Myrtveit. « A simulation study of the model evaluation criterion MMRE ». In : *IEEE Transactions on Software Engineering* 29.11 (nov. 2003), p. 985–995 (cf. p. 31, 35).

[FSF2015] FSF. *gnu.org*. 2015 (cf. p. 90).

[Galorath+2006] Daniel D. Galorath et Michael W. Evans. *Software Sizing, Estimation, and Risk Management: When Performance is Measured Performance Improves*. en. CRC Press, jan. 2006 (cf. p. 6, 18).

[Garavaglia+1998] S. Garavaglia et A. Sharma. « A smart guide to dummy variables: four applications and a macro ». In : *Proceedings of the Northeast SAS Users Group Conference*. Citeseer, 1998 (cf. p. 43).

[Glass2006] Robert L. Glass. « The Standish report: does it really describe a software crisis? » en. In : *Communications of the ACM* 49.8 (août 2006), p. 15 (cf. p. 4).

[Goodman1992] PA Goodman. « Application of cost-estimation techniques: industrial perspective ». In : *Information and Software Technology* 34.6 (juin 1992), p. 379–382 (cf. p. 15).

[Guyon+2003] Isabelle Guyon et André Elisseeff. « An introduction to variable and feature selection ». In : *The Journal of Machine Learning Research* 3 (2003), p. 1157–1182 (cf. p. 42).

[Hannay+2007] Jo Erskine Hannay, Dag IK Sjoberg et Tore Dyba. « A systematic review of theory use in software engineering experiments ». In : *Software Engineering, IEEE Transactions on* 33.2 (2007), p. 87–107 (cf. p. 22).

[Hardin+2005] Johanna Hardin et David M Rocke. « The Distribution of Robust Distances ». en. In : *Journal of Computational and Graphical Statistics* 14.4 (déc. 2005), p. 928–946 (cf. p. 41).

[Hauke+2011] Jan Hauke et Tomasz Kossowski. « Comparison of Values of Pearson's and Spearman's Correlation Coefficients on the Same Sets of Data ». In : *Quaestiones Geographicae* 30.2 (jan. 2011) (cf. p. 42).

[Herrador+2005] M. Angeles Herrador, Agustin G. Asuero et A. Gustavo González. « Estimation of the uncertainty of indirect measurements from the propagation of distributions by using the Monte-Carlo method: An overview ». In : *Chemometrics and Intelligent Laboratory Systems* 79.1–2 (oct. 2005), p. 115–122 (cf. p. 29, 63).

[Hird+2015] Abigail Hird, Kepa Mendibil, Alex Duffy et Robert Ian Whitfield. « New product development resource forecasting ». en. In : *R&D Management* (juin 2015), n/a–n/a (cf. p. 22).

[Holm2011] M. Holm. « Construction and evaluation of a tool for quantifying uncertainty of software cost estimates ». Thèse de doct. Norway : University of OSLO, 2011 (cf. p. 26).

[Hopfield1982] J. J. Hopfield. « Neural networks and physical systems with emergent collective computational abilities. » en. In : *Proceedings of the National Academy of Sciences* 79.8 (avr. 1982), p. 2554–2558 (cf. p. 19).

[Huang+2007] Sun-Jen Huang et Nan-Hsing Chiu. « Applying fuzzy neural network to estimate software development effort ». In : *Applied Intelligence* 30.2 (oct. 2007), p. 73–83 (cf. p. 26).

[Humphrey2001] Watts S. HUMPHREY. *Winning with Software: An Executive Strategy*. en. Pearson Education, déc. 2001 (cf. p. 2, 4).

[Idri+2010] A. IDRI, Abdelali ZAKRANI et Azeddine ZAHI. « Design of radial basis function neural networks for software effort estimation ». In : *IJCSI International Journal of Computer Science Issues* 7.4 (2010), p. 11–17 (cf. p. 20).

[Idri+2015] Ali IDRI, Fatima azzahra AMAZAL et Alain ABRAN. « Analogy-based software development effort estimation: A systematic mapping and review ». In : *Information and Software Technology* 58 (fév. 2015), p. 206–230 (cf. p. 34).

[IFPUG1999] IFPUG. *Function Point Counting Practices Manual*. 1999 (cf. p. 13).

[International2013] Standish Group INTERNATIONAL. *Chaos Manifesto 2013*. Rapp. tech. 2013 (cf. p. 1).

[ISO2003] ISO. *ISO/IEC 19761:2003 - Software engineering – COSMIC-FFP – A functional size measurement method*. 2003 (cf. p. 13).

[ISO2007] ISO. *Information technology – Software measurement – Functional size measurement – Part 1: Definition of concepts*. Rapp. tech. 14143-1:2007. 2007 (cf. p. 12).

[JCGM2008] JCGM. *Évaluation des données de mesure — Guide pour l'expression de l'incertitude de mesure*. 2008 (cf. p. 24, 25, 61).

[Jorgensen+2002] Magne JORGENSEN et Kjetil MOLØKKEN. « Combination of software development effort prediction intervals: why, when and how? » In : *Proceedings of the 14th international conference on Software engineering and knowledge engineering*. 2002, p. 425–428 (cf. p. 27).

[Jorgensen+2003] M. JORGENSEN et Dag I. K. SJOEBERG. « An effort prediction interval approach based on the empirical distribution of previous estimation accuracy ». In : *Information and Software Technology* 45.3 (2003), p. 123–136 (cf. p. 27, 96).

[Jorgensen+2004a] M. JORGENSEN, K. H. TEIGEN et K. MOLØKKEN. « Better sure than safe? Overconfidence in judgement based software development effort prediction intervals ». In : *Journal of Systems and Software* 70.1-2 (fév. 2004), p. 79–93 (cf. p. 30).

[Jorgensen+2004b] Magne JORGENSEN et Kjetil MOLOKKEN. « Eliminating over-confidence in software development effort estimates ». In : *Product Focused Software Process Improvement*. Springer, 2004, p. 174–184 (cf. p. 28).

[Jorgensen+2007] Magne JORGENSEN et Martin SHEPPERD. « A systematic review of software development cost estimation studies ». In : *Software Engineering, IEEE Transactions on* 33.1 (2007), p. 33–53 (cf. p. 13).

[Jorgensen1995] Magne JORGENSEN. « Experience With the Accuracy of Software Maintenance Task Effort Prediction Models ». In : *IEEE Trans. Softw. Eng.* 21.8 (août 1995), p. 674–681 (cf. p. 31).

[Jorgensen2004a] M. JORGENSEN. « A review of studies on expert estimation of software development effort ». In : *Journal of Systems and Software* 70.1-2 (2004), p. 37–60 (cf. p. 15).

[Jorgensen2004b] M. JORGENSEN. « Top-down and bottom-up expert estimation of software development effort ». In : *Information and Software Technology* 46.1 (2004), p. 3–16 (cf. p. 15).

[Jorgensen2007] M. JORGENSEN. « Forecasting of software development work effort: evidence on expert judgement and formal models ». In : *International Journal of Forecasting* 23.3 (2007), p. 449–462 (cf. p. 14).

[Kaushik+2012] Anupama KAUSHIK, Ashish CHAUHAN, Deepak MITTAL et Sachin GUPTA. « COCOMO Estimates Using Neural Networks ». In : *International Journal of Intelligent Systems and Applications* 4.9 (août 2012), p. 22–28 (cf. p. 20).

[Keung+2012] Jacky KEUNG, Ekrem KOCAGUNELI et Tim MENZIES. « Finding conclusion stability for selecting the best effort predictor in software effort estimation ». In : *Automated Software Engineering* 20.4 (mai 2012), p. 543–567 (cf. p. 34).

[Keung2009] Jacky KEUNG. « Software development cost estimation using analogy: a review ». In : *Software Engineering Conference, 2009. ASWEC'09. Australian*. 2009, 327â€“336 (cf. p. 15).

[Kitchenham+1997] B. KITCHENHAM et S. LINKMAN. « Estimates, uncertainty, and risk ». In : *IEEE Software* 14.3 (mai 1997), p. 69–74 (cf. p. 25, 26).

[Kitchenham+1999] Barbara KITCHENHAM, Stephen MACDONELL, Lesley PICKARD et Martin SHEPPERD. « Assessing prediction systems ». In : *IEEE Transactions on Software Engineering - TSE* (1999) (cf. p. 31, 55).

[Kitchenham+2001] Barbara A. KITCHENHAM, Lesley M. PICKARD, Stephen G. MACDONELL et Martin J. SHEPPERD. « What accuracy statistics really measure [software estimation] ». In : *Software, IEE Proceedings-*. T. 148. IET, 2001, p. 81–85 (cf. p. 32, 35).

[Kitchenham+2003] B. KITCHENHAM, L.M. PICKARD, S. LINKMAN et P.W. JONES. « Modeling software bidding risks ». In : *IEEE Transactions on Software Engineering* 29.6 (juin 2003), p. 542–554 (cf. p. 26).

[Klas+2011] Michael KLAS, Adam TRENDOWICZ, Yasushi ISHIGAI et Haruka NAKAO. « Handling estimation uncertainty with bootstrapping: Empirical evaluation in the context of hybrid prediction methods ». In : *Empirical Software Engineering and Measurement (ESEM), 2011 International Symposium on*. 2011, p. 245–254 (cf. p. 28).

[Kohavi1995] Ron KOHAVI. « A study of cross-validation and bootstrap for accuracy estimation and model selection ». In : *International joint Conference on artificial intelligence*. T. 14. 1995, p. 1137–1145 (cf. p. 73).

[Krstajic+2014] Damjan KRSTAJIC, Ljubomir J. BUTUROVIC, David E. LEAHY et Simon THOMAS. « Cross-validation pitfalls when selecting and assessing regression and classification models ». In : *Journal of cheminformatics* 6.1 (2014), p. 1–15 (cf. p. 73).

[LaGrega+2010] Michael D. LAGREGA, Phillip L. BUCKINGHAM et Jeffrey C. EVANS. *Hazardous Waste Management*. English. Reissue edition. Long Grove, Ill. : Waveland Pr Inc, juil. 2010 (cf. p. 29).

[Laqrichi+2013a] Safae LAQRICHI, Francois MARMIER et Didier GOURC. « Proposition d'une classification des méthodes d'estimation des projets de systèmes d'information ». In : *Congrés International de Génie Industriel CIGI2013*. 2013 (cf. p. 13).

[Laqrichi+2013b] Safae LAQRICHI, François MARMIER et Didier GOURC. « Toward an effort estimation model for information system project integrating risk ». In : *Proc. 22nd International Conference on Production Research (ICPR22)* (2013) (cf. p. 99).

[Laqrichi+2015] Safae LAQRICHI, Francois MARMIER et DIDIER GOURC. « Integrating uncertainty in effort estimation using Bootstrap based Neural Networks ». In : (2015) (cf. p. 28).

[Lemberger+2015] Pirmin LEMBERGER, Marc BATTY, Médéric MOREL et Jean-Luc RAFFAËLLI. *Big Data et machine learning: Manuel du data scientist*. fr. Dunod, fév. 2015 (cf. p. 42).

[Li+2010] Xiang LI, Min XIE et Szu Hui NG. « Sensitivity analysis of release time of software reliability models incorporating testing effort with multiple change-points ». In : *Applied Mathematical Modelling* 34.11 (nov. 2010), p. 3560–3570 (cf. p. 28).

[LopezMartin2015] Cuauhtémoc LOPEZ-MARTIN. « Predictive accuracy comparison between neural networks and statistical regression for development effort of software projects ». In : *Applied Soft Computing* 27 (fév. 2015), p. 434–449 (cf. p. 14).

[MacCallum+1986] K. J. MACCALLUM et J.-M. ZHANG. « Curve-smoothing Techniques Using B-splines ». en. In : *The Computer Journal* 29.6 (jan. 1986), p. 564–571 (cf. p. 66).

[Mair+2005] Carolyn MAIR et Martin SHEPPERD. « The consistency of empirical comparisons of regression and analogy-based software project cost prediction ». In : *Empirical Software Engineering, 2005. 2005 International Symposium on*. IEEE, 2005, 10–pp (cf. p. 34).

[Marmier+2013] F. MARMIER, D. GOURC et F. LAARZ. « A risk oriented model to assess strategic decisions in new product development projects ». In : *Decision Support Systems* 56 (déc. 2013), p. 74–82 (cf. p. 98).

[Masateru Tsunoda2010] Akito Monden MASATERU TSUNODA. « Applying Outlier Deletion to Analosgy Based Cost Estimation ». In : (2010) (cf. p. 41).

[McConnell2006] Steve MCCONNELL. *Software estimation: demystifying the black art*. Redmond, Wash : Microsoft Press, 2006 (cf. p. 5, 26).

[Menzies+2012] Tim MENZIES et Martin SHEPPERD. « Special issue on repeatable results in software engineering prediction ». en. In : *Empirical Software Engineering* 17.1-2 (fév. 2012), p. 1–17 (cf. p. 34).

[Mitchell1997] Tom M. MITCHELL. *Machine Learning*. McGraw-Hill series in computer science. New York : McGraw-Hill, 1997 (cf. p. 21).

[Mittas+2009] Nikolaos MITTAS et Lefteris ANGELIS. « Bootstrap Confidence Intervals for Regression Error Characteristic Curves Evaluating the Prediction Error of Software Cost Estimation Models. » In : *AIAI Workshops*. 2009, p. 221–230 (cf. p. 27, 28).

[Mittas+2010] Nikolaos MITTAS et Lefteris ANGELIS. « LSEbA: least squares regression and estimation by analogy in a semi-parametric model for software cost estimation ». en. In : *Empirical Software Engineering* 15.5 (oct. 2010), p. 523–555 (cf. p. 16, 97).

[Mittas+2015a] Nikolaos MITTAS, Ioannis MAMALIKIDIS et Lefteris ANGELIS. « A framework for comparing multiple cost estimation methods using an automated visualization toolkit ». In : *Information and Software Technology* 57 (jan. 2015), p. 310–328 (cf. p. 34).

[Mittas+2015b] Nikolaos MITTAS, Efi PAPATHEOCHAROUS, Lefteris ANGELIS et Andreas S. ANDREOU. « Integrating non-parametric models with linear components for producing software cost estimations ». In : *Journal of Systems and Software* 99 (jan. 2015), p. 120–134 (cf. p. 97).

[Mittas2011] N. MITTAS. « Evaluating the Performances of Software Cost Estimation Models through Prediction Intervals ». In : *Journal of Engineering Science and Technology Review* 4.3 (2011), p. 266–270 (cf. p. 27).

[Miyazaki+1994] Y. MIYAZAKI, M. TERAKADO, K. OZAKI et H. NOZAKI. « Robust regression for developing software estimation models ». In : *Journal of Systems and Software* 27.1 (oct. 1994), p. 3–16 (cf. p. 30, 32).

[Moder+1995] Joseph J. MODER, Cecil R. PHILLIPS et Edward W. DAVIS. *Project Management With Cpm, Pert & Precedence Diagramming*. English. 3rd edition. New York : Blitz Publishing Company, fév. 1995 (cf. p. 26).

[MoløkkenØstvold+2008] Kjetil MOLØKKEN-ØSTVOLD, Nils Christian HAUGEN et Hans Christian BENESTAD. « Using planning poker for combining expert estimates in software projects ». In : *Journal of Systems and Software* 81.12 (déc. 2008), p. 2106–2117 (cf. p. 15).

[Moody+1995] J. MOODY, S. HANSON, Anders KROGH et John A. HERTZ. « A simple weight decay can improve generalization ». In : *Advances in neural information processing systems* 4 (1995), p. 950–957 (cf. p. 49).

[Mooney1997] Christopher Z. MOONEY. *Monte Carlo Simulation*. en. SAGE Publications, avr. 1997 (cf. p. 61).

[Moores2001] T.T. MOORES. « Developing a software size model for rule-based systems: a case study ». In : *Expert Systems with Applications* 21.4 (nov. 2001), p. 229–237 (cf. p. 12).

[Muzaffar+2010] Zeeshan MUZAFFAR et Moataz A. AHMED. « Software development effort prediction: A study on the factors impacting the accuracy of fuzzy logic systems ». In : *Information and Software Technology* 52.1 (jan. 2010), p. 92–109 (cf. p. 13).

[Myrtveit+2005] Ingunn MYRTVEIT, Erik STENSRUD et Martin SHEPPERD. « Reliability and validity in comparative studies of software prediction models ». In : *Software Engineering, IEEE Transactions on* 31.5 (2005), p. 380–391 (cf. p. 31).

[Nassif+2011] Ali Bou NASSIF, Danny HO et Luiz Fernando CAPRETZ. « Regression Model for Software Effort Estimation Based on the Use Case Point Method ». In : (2011) (cf. p. 18).

[Nassif+2013] Ali Bou NASSIF, Mohammad AZZEH, Luiz Fernando CAPRETZ et Danny HO. « A comparison between decision trees and decision tree forest models for software development effort estimation ». In : *Communications and Information Technology (ICCIT), 2013 Third International Conference on*. IEEE, 2013, p. 220–224 (cf. p. 21).

[Nguyen+2013] Trong-Hung NGUYEN, Francois MARMIER et Didier GOURC. « A decision-making tool to maximize chances of meeting project commitments ». In : *International Journal of Production Economics* 142.2 (avr. 2013), p. 214–224 (cf. p. 98).

[Papadopoulos+2001] G. PAPADOPOULOS, P.J. EDWARDS et A.F. MURRAY. « Confidence estimation methods for neural networks: a practical comparison ». In : *IEEE Transactions on Neural Networks* 12.6 (nov. 2001), p. 1278–1287 (cf. p. 27).

[Pfleeger2005] Shari Lawrence PFLEEGER. *Software Cost Estimation and Sizing Methods, Issues, and Guidelines*. Rand Publishing, sept. 2005 (cf. p. 13).

[Planchon2005] Viviane PLANCHON. « Traitement des valeurs aberrantes: concepts actuels et tendances générales ». In : *Biotechnologie, agronomie, société et environnement* 9.1 (2005), p. 19–34 (cf. p. 41).

[Port+2008] Dan PORT et Marcel KORTE. « Comparative studies of the model evaluation criterions mmre and pred in software cost estimation research ». In : *Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement*. ACM, 2008, p. 51–60 (cf. p. 28).

[Putnam1978] Lawrence H. PUTNAM. « A general empirical solution to the macro software sizing and estimating problem ». In : *IEEE transactions on Software Engineering* 4 (1978), p. 345–361 (cf. p. 17).

[Razali+2011] Nornadiah Mohd RAZALI et Yap Bee WAH. « Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests ». In : *Journal of Statistical Modeling and Analytics* 2.1 (2011), p. 21–33 (cf. p. 43, 67).

[Reddy+2010] Ch. Satyananda REDDY et R. KVSVN. « An Optimal Neural Network Model for Software Effort Estimation ». In : *Int. J. of Software Engineering, IJSE* 3.1 (2010) (cf. p. 20).

[Refaeilzadeh+2009] Payam REFAEILZADEH, Lei TANG et Huan LIU. « Cross-Validation ». In : *Encyclopedia of Database Systems*. Sous la dir. de Ling LIU et Tamer ÖZSU. Springer US, 2009, p. 532–538 (cf. p. 33).

[Reunanen2003] Juha REUNANEN. « Overfitting in making comparisons between variable selection methods ». In : *The Journal of Machine Learning Research* 3 (2003), p. 1371–1382 (cf. p. 35).

[Sadiq2013] Mohd SADIQ. « Prediction of Software Project Effort Using LinearRegression Model ». In : *International Journal of Information and Electronics Engineering* (2013) (cf. p. 18).

[Saeys+2007] Yvan SAEYS, Iñaki INZA et Pedro LARRAÑAGA. « A review of feature selection techniques in bioinformatics ». en. In : *Bioinformatics* 23.19 (jan. 2007), p. 2507–2517 (cf. p. 42).

[Sakia1992] R. M. SAKIA. « The Box-Cox Transformation Technique: A Review ». In : *Journal of the Royal Statistical Society. Series D (The Statistician)* 41.2 (jan. 1992), p. 169–178 (cf. p. 43).

[Saltelli+2009] A. SALTELLI, K. CHAN et E. M. SCOTT. *Sensitivity Analysis*. English. 1 edition. Chichester : Wiley, mar. 2009 (cf. p. 28).

[Shapiro+1965] S. S. SHAPIRO et M. B. WILK. « An Analysis of Variance Test for Normality (Complete Samples) ». In : *Biometrika* 52.3/4 (déc. 1965), p. 591 (cf. p. 67).

[Shashank Mouli Satapathy2014] Barada Prasanna Acharya SHASHANK MOULI SATAPATHY. « Early Stage Software Effort Estimation using Random Forest Technique based on Optimized Class Point Approach ». In : *INFOCOMP Journal of Computer Science* 13.2 (2014), p. 22–33 (cf. p. 21).

[Shepperd+1997] M. SHEPPERD et C. SCHOFIELD. « Estimating software project effort using analogies ». In : *IEEE Transactions on Software Engineering* 23.11 (nov. 1997), p. 736–743 (cf. p. 15).

[Shepperd+2001] M. SHEPPERD et G. KADODA. « Comparing software prediction techniques using simulation ». In : *IEEE Transactions on Software Engineering* 27.11 (nov. 2001), p. 1014–1022 (cf. p. 34).

[Shepperd+2012] Martin SHEPPERD et Steve MACDONELL. « Evaluating prediction systems in software project estimation ». In : *Information and Software Technology*. Special Issue: Voice of the Editorial Board Special Issue: Voice of the Editorial Board 54.8 (août 2012), p. 820–827 (cf. p. 31).

[Statnikov+2005] Alexander STATNIKOV, Constantin F. ALIFERIS, Ioannis TSAMARDINOS, Douglas HARDIN et Shawn LEVY. « A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis ». en. In : *Bioinformatics* 21.5 (jan. 2005), p. 631–643 (cf. p. 47).

[Suma+2014] V. SUMA, T. P. PUSHPHAVATHI et V. RAMASWAMY. « An Approach to Predict Software Project Success Based on Random Forest Classifier ». en. In : *ICT and Critical Infrastructure: Proceedings of the 48th Annual Convention of Computer Society of India- Vol II*. Sous la dir. de Suresh Chandra SATAPATHY, P. S. AVADHANI, Siba K. UDGATA et Sadasivuni LAKSHMINARAYANA. Advances in Intelligent Systems and Computing 249. Springer International Publishing, jan. 2014, p. 329–336 (cf. p. 21).

[Sunkle+2012] Sagar SUNKLE et Vinay KULKARNI. « Cost Estimation for Model-Driven Engineering ». en. In : *Model Driven Engineering Languages and Systems*. Sous la dir. de Robert B. FRANCE, Jürgen KAZMEIER, Ruth BREU et Colin ATKINSON. Lecture Notes in Computer Science 7590. Springer Berlin Heidelberg, 2012, p. 659–675 (cf. p. 13).

[Surendiran+2010] B. SURENDIRAN et A. VADIVEL. « Feature selection using stepwise ANOVA discriminant analysis for mammogram mass classification ». In : *International J. of Recent Trends in Engineering and Technology* 3.2 (2010), p. 55–57 (cf. p. 42).

[Trendowicz2014] Adam TRENDOWICZ. *Software Project Effort Estimation: Foundations and Best Practice Guidelines for Success*. en. Springer, jan. 2014 (cf. p. 32, 34).

[Tufféry2010] Stéphane TUFFÉRY. *Data mining et statistique décisionnelle: l'intelligence des données*. fr. Editions TECHNIP, jan. 2010 (cf. p. 20).

[Urbanek2003] Simon URBANEK. « Rserve – A Fast Way to Provide R Functionality to Applications ». In : *PROC. OF THE 3RD INTERNATIONAL WORKSHOP ON DISTRIBUTED STATISTICAL COMPUTING (DSC 2003), ISSN 1609-395X, EDS.: KURT HORNIK, FRIEDRICH LEISCH & ACHIM ZEILEIS, 2003 (HTTP://ROSUDA.ORG/RSERVE*. 2003 (cf. p. 89).

[Varma+2006] Sudhir VARMA et Richard SIMON. « Bias in error estimation when using cross-validation for model selection ». In : *BMC bioinformatics* 7.1 (2006), p. 91 (cf. p. 35, 46, 47).

[Wallace+2004] Linda WALLACE, Mark KEIL et Arun RAI. « How Software Project Risk Affects Project Performance: An Investigation of the Dimensions of Risk and an Exploratory Model* ». en. In : *Decision Sciences* 35.2 (2004), 289â€“321 (cf. p. 99).

[Wen+2012] Jianfeng WEN, Shixian LI, Zhiyong LIN, Yong HU et Changqin HUANG. « Systematic literature review of machine learning based software development effort estimation models ». In : *Information and Software Technology* 54.1 (jan. 2012), p. 41–59 (cf. p. 34).

[Yitzhaki2003] Shlomo YITZHAKI. « Gini's mean difference: A superior measure of variability for non-normal distributions ». In : *Metron* 61.2 (2003), p. 285–316 (cf. p. 54).

[Yu+2003] Lei YU et Huan LIU. « Feature selection for high-dimensional data: A fast correlation-based filter solution ». In : *ICML*. T. 3. 2003, p. 856–863 (cf. p. 42).

RÉSUMÉ

Approche pour la construction de modèles d'estimation réaliste de l'effort/coût de projet dans un environnement incertain : application au domaine du développement logiciel

L'estimation de l'effort de développement logiciel est l'une des tâches les plus importantes dans le management de projets logiciels. Elle constitue la base pour la planification, le contrôle et la prise de décision. La réalisation d'estimations fiables en phase amont des projets est une activité complexe et difficile du fait, entre autres, d'un manque d'informations sur le projet et son avenir, de changements rapides dans les méthodes et technologies liées au domaine logiciel et d'un manque d'expérience avec des projets similaires. De nombreux modèles d'estimation existent, mais il est difficile d'identifier un modèle performant pour tous les types de projets et applicable à toutes les entreprises (différents niveaux d'expérience, technologies maitrisées et pratiques de management de projet). Globalement, l'ensemble de ces modèles formule l'hypothèse forte que (1) les données collectées sont complètes et suffisantes, (2) les lois reliant les paramètres caractérisant les projets sont parfaitement identifiables et (3) que les informations sur le nouveau projet sont certaines et déterministes. Or, dans la réalité du terrain cela est difficile à assurer. Deux problématiques émergent alors de ces constats : comment sélectionner un modèle d'estimation pour une entreprise spécifique ? et comment conduire une estimation pour un nouveau projet présentant des incertitudes ? Les travaux de cette thèse s'intéressent à répondre à ces questions en proposant une approche générale d'estimation. Cette approche couvre deux phases : une phase de construction du système d'estimation et une phase d'utilisation du système pour l'estimation de nouveaux projets. La phase de construction du système d'estimation est composée de trois processus : 1) évaluation et comparaison fiable de différents modèles d'estimation, et sélection du modèle d'estimation le plus adéquat, 2) construction d'un système d'estimation réaliste à partir du modèle d'estimation sélectionné et 3) utilisation du système d'estimation dans l'estimation d'effort de nouveaux projets caractérisés par des incertitudes. Cette approche intervient comme un outil d'aide à la décision pour les chefs de projets dans l'aide à l'estimation réaliste de l'effort, des coûts et des délais de leurs projets logiciels. L'implémentation de l'ensemble des processus et pratiques développés dans le cadre de ces travaux ont donnée naissance à un prototype informatique open-source. Les résultats de cette thèse s'inscrivent dans le cadre du projet ProjEstimate FUI13.

MOTS-CLÉS : Sélection de modèle d'estimation, Incertitude, Validation croisée, Réseaux de neurones, Forêts d'arbres décisionnels, Bootstrap

---

ABSTRACT

Approach to build realistic models for estimating project effort/cost in an uncertain environment : application to the software development field

Software effort estimation is one of the most important tasks in the management of software projects. It is the basis for planning, control and decision making. Achieving reliable estimates in projects upstream phases is a complex and difficult activity because, among others, of the lack of information about the project and its future, the rapid changes in the methods and technologies related to the software field and the lack of experience with similar projects. Many estimation models exist, but it is difficult to identify a successful model for all types of projects and that is applicable to all companies (different levels of experience, mastered technologies and project management practices). Overall, all of these models form the strong assumption that (1) the data collected are complete and sufficient, (2) laws linking the parameters characterizing the projects are fully identifiable and (3) information on the new project are certain and deterministic. However, in reality on the ground, that is difficult to be ensured. Two problems then emerge from these observations: how to select an estimation model for a specific company? and how to conduct an estimate for a new project that presents uncertainties? The work of this thesis interested in answering these questions by proposing a general estimation framework. This framework covers two phases: the construction phase of the estimation system and system usage phase for estimating new projects. The construction phase of the rating system consists of two processes: 1) evaluation and reliable comparison of different estimation models then selection the most suitable estimation model, 2) construction of a realistic estimation system from the selected estimation model and 3) use of the estimation system in estimating effort of new projects that are characterized by uncertainties. This approach acts as an aid to decision making for project managers in supporting the realistic estimate of effort, cost and time of their software projects. The implementation of all processes and practices developed as part of this work has given rise to an open-source computer prototype. The results of this thesis fall in the context of ProjEstimate FUI13 project.

KEYWORDS: Estimation model selection, Uncertitudy, Cross validation, Neural networks, Random forests, Bootstrap